Classifying scanners:
Mapping their behaviour

V.D.H. Ghiëtte

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

CLASSIFYING SCANNERS:

MAPPING THEIR BEHAVIOUR

by

V.D.H. Ghiëtte

in partial fulfillment of the requirements for the degree of

Master of Science
in Computer Science

at the Delft University of Technology,
to be defended publicly on Thursday November 3 2016 at 14:00.

Supervisor:	Dr. Christian Doerr
Thesis committee:	Prof. Dr. Jan van den Berg
	Dr. Christian Doerr
	Dr. Johan Pouwelse

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft Delft University of Technology

PREFACE

Before you lies the thesis "Classifying scanners: Mapping their behaviour" written to fulfill the graduation requirements of the Computer science program at the Delft University of Technology.

I would like to thank my supervisor Dr. Christian Doerr, and Dr. Norbert Blenn for their excellent guidance and support during this process.

V.D.H. Ghiëtte
Delft, September 2016

ABSTRACT

This thesis focuses on the classification of behavioural aspects of scanners based on unroutable traffic collected from two /16 subnets. Firstly the study determines that the use of a smaller dataset achieves similar results and allows for the same correctness compared to larger ones. Secondly different scanning tools are analysed, and methods for their fingerprinting are explained. The implementation of detection methods reveals the usage of particular tools and the the existence of previously unknown software. Analysing these previously unknown tools shows that there is a difference in levels of sophistication of the tools used by scanners. Following, this thesis confirms the existence of the horizontal, vertical and strobe scanner classes, it also describes a new method in which a destination port and address are only scanned once. In addition a method to identify individual scans from traffic captures for further analysis is presented and evaluated. This method is then used to reveal similarities between scans from one address but also confirming collaboration between multiple addresses. Finally the behaviour of scanners is compared showing cyclic behaviours are common amongst single and multi host scanners.

Contents

List of Figures ix

List of Tables xi

1 Introduction 1

2 Background information 3
2.1 Communication methods . . . 3
2.1.1 The Open System Interconnections Reference Model (OSI) . . . 3
2.1.2 The Network layer . . . 3
2.1.3 The Transport Layer . . . 6
2.2 Topology of the Internet . . . 8

3 Related work 11

4 Data collection 15
4.1 Data provenance . . . 15
4.2 Basic analysis . . . 15
4.3 Data sanitation . . . 17
4.3.1 Network layer sanitation . . . 17
4.3.2 Transport layer sanitation . . . 17
4.3.3 Subnet sanitation . . . 17
4.4 Challenges . . . 18

5 Data analysis based on used protocol 19
5.1 Targeted destination addresses . . . 19
5.2 UDP . . . 20
5.2.1 Targeted destination addresses . . . 20
5.2.2 Destination ports . . . 21
5.2.3 Source ports . . . 22
5.3 TCP . . . 22
5.3.1 Targeted destination addresses (TCP) . . . 22
5.3.2 Type of packets . . . 23
5.3.3 Targeted destination addresses (TCP SYN) . . . 23
5.3.4 Destination ports . . . 23
5.3.5 Source ports . . . 24
5.3.6 Other TCP header values . . . 24
5.3.7 IPv4 header values . . . 25

6 Identification of scanning tools 27
6.1 Known scanning techniques . . . 27
6.1.1 ICMP . . . 27
6.1.2 UDP . . . 27
6.1.3 TCP . . . 28
6.2 Known scanners . . . 28
6.2.1 ZMap . . . 28
6.2.2 Masscan . . . 29
6.2.3 Unicornscan . . . 29
6.2.4 NMap . . . 30

6.3 Confidence levels and intervals . . . 31
6.4 ZMap like . . . 32
6.5 Masscan like . . . 33
6.6 Unicornscan Like . . . 34
6.7 NMap like . . . 34

7 Spacial analysis **35**
7.1 Origin of scans . . . 35
7.2 Countries . . . 36
7.2.1 Number of addresses . . . 36
7.2.2 Number of TCP SYN packets . . . 39
7.2.3 Used scanning tools . . . 40
7.3 Which autonomous systems are more active . . . 40

8 Time analysis **43**
8.1 Daily analysis . . . 43
8.2 Hourly analysis . . . 44
8.3 Scan time analysis . . . 46

9 Scanner behaviours **49**
9.1 Single source scanners . . . 49
9.1.1 General overview . . . 49
9.1.2 Particular behaviour . . . 51
9.2 Scanners using multiple machines or source IP . . . 54
9.2.1 Known scanners . . . 54
9.2.2 Unknown scanners . . . 56

10 Final thoughts **59**
10.1 Discussion . . . 59
10.2 Conclusion . . . 61
10.3 Future work . . . 62

A Packet type received per address **63**

Bibliography **65**

List of Figures

2.1 The OSI reference model . . . 4
2.2 IPv4 Header as presented by Jon Postel in the Internet Protocol RFC [1] . . . 5
2.3 The IPv4 binary to decimal conversion . . . 5
2.4 IP range calculation using slash notation . . . 6
2.5 UDP Header as presented by J. Postel in RFC 768 [2] . . . 6
2.6 TCP Header, defined in RFC 793 [3] . . . 7
2.7 The TCP three way handshake . . . 8

5.1 Distribution of number of packets received per IPv4 destination address . . . 19
5.2 Number of UDP packets received per port . . . 21
5.3 Number of TCP SYN packets received per Port . . . 23
5.4 Distribution of received acknowledgement numbers . . . 25
5.5 Distribution of IPv4 header values . . . 26

6.1 NMap TCP sequence number generator . . . 30
6.2 NMap TCP source port generation . . . 30
6.3 Lower confidence bound per confidence level for given number of samples . . . 32

7.1 Illustration of the Hilber Curver for address spaces of 2,3, and 4 bits. . . . 36
7.3 Geographical location of IPv4 addresses having sent TCP SYN packets . . . 36
7.2 IPv4 heatmap of IPv4 having sent packets . . . 37
7.4 Geographical location of IPv4 addresses having sent ZMap like TCP SYN packets . . . 40
7.5 Number of packets and IPv4 addresses per AS . . . 41

8.1 Normalized number of scanners and received packets per day. . . . 43
8.2 Distribution of number of scanners by number of days scanned . . . 44
8.3 Normalized number of scanners and received packets per day. . . . 44
8.4 Distribution of time intervals between packets . . . 46
8.5 Distribution of scan duration . . . 47

9.1 Unique addresses and address/port scanned per source address . . . 50
9.2 Plot of scanners showing the umber of unique IP and unique ports scanned . . . 50
9.3 CDF of number of packets sent per source IP of cluster 43.255.191.141.0/24 . . . 56
9.4 Plot of scanners showing the umber of unique IP and unique ports scanned . . . 57

A.1 Number of packets received per IPv4 destination address . . . 63

List of Tables

4.2 Number of packages received from each transport layer protocol type (percentage) 16
4.1 Number of frames received from each network protocol type (percentage) 16

5.1 Top 10 destination ports, in terms of received UDP packets (percentage) *Service specified by IANA [4] **Netis System Backdoor ***ASF Remote Management and Control Protocol 21
5.2 Top 5 source ports, in terms of sending UDP packets (percentage) 22
5.3 Top 3 set flags in terms of received TCP packets (percentage) 22
5.5 Top 10 destination ports, in terms of received TCP SYN packets (percentage) * Mainly Tomcat server **Likely port used for setting up a proxy 24
5.4 Top 5 source ports, in terms of sending TCP SYN packets (percentage) 24
5.6 Top 5 sequence numbers, in terms of sending TCP SYN packets (percentage) 25

6.1 ZMap like scanner per ip id 33

7.1 Top 10 countries showing scanning behaviour * Percentage of IPv4 addresses of a country ***Number of addresses used divided by the population 38
7.2 Top 10 AS related to the IPv4 having sent TCP SYN packets 41
7.3 Top 5 AS sent TCP SYN packets 41

9.1 Percentage of packets received on destination ports classified by used tools 51
9.2 Port scanned by 180.97.192.16 52
9.3 Port scanned by 146.185.239.104 52
9.4 Known ZMap scanners having IP identification number set to 54321 55
9.5 Port scanned by 94.31.49.0/24 57
9.6 Port scanned by 115.160.65.105/24 58

1

INTRODUCTION

The Internet is a large network of independent networks of devices. Recently, the last block of available Internet Protocol version 4 (IPv4) addresses has been allocated by the Internet Corporation for Assigned Names and Numbers [5]. This last allocation reveals the sheer size of the Internet, which has reached the maximum amount of IPv4 addresses possible, 3.6 billion. The address range offers a lower bound on the size of the Internet, as the Internet is a network of independent networks. Multiple devices belonging to an independent network may reside behind one Internet address using network address translation (NAT), an estimate made by Juniper Research [6] suggest that the actual number of devices connected to the internet is in the order of 10 billion. Heidemann et al. performed research [7] resulting in a census of the Internet. The results suggest that only 3.6% of all addresses can be classified as reachable meaning that the devices at these addresses respond to messages. More recent research, dating from 2015 [8], implies that the reachable part of the internet has grown to 10.4%. This amount of reachable devices seems low as all of the IPv4 addresses are allocated. However some addresses are not assigned to devices permanently connected to the Internet, and some addresses are allocated but not yet assigned to devices. Most likely the probed devices did not respond to the ping messages sent during the analysis.

The results presented in the mentioned research are based on data retrieved by sending Internet Control Message Protocol (ICMP) messages to all addresses in the IPv4 range and counting the number of received responses to those messages. This technique of sending ICMP messages and monitoring the responses is called pinging the Internet, it allows for detecting active devices on the Internet. The result of this activity is a list containing all the responding IPv4 addresses, providing the scanner information about which devices are connected to the Internet. However this information has its restrictions, the ICMP does not offer information about the the type of services the machines on the scanned addresses are providing or whether there are devices behind Network Address Translation routers etc.

Therefore a more powerful technique of revealing information about connected devices to the Internet is required called port scanning. Port scanning not only provides a list of addresses, but also a list of open ports for each address, giving an indication of the services hosted on the machine. Recently progress has been made in the field of port scanning, new tools such as NMap [9] and ZMap [10], which enable fast scanning of services on the Internet were developed. Study performed by Durumeric et al. [10] reveals that ZMap can scan the entire Internet within hours providing fast and up-to-date information about services offered on the Internet. This data can be used for research purposes to get a better understanding of the services offered on the Internet, or to project the dynamics of the Internet. However such data can also hold value for malicious parties.

The services offered on the Internet not always use secure protocols, or have faulty implementations rendering them insecure. Such bugs are not rare, a recently discovered security bug, ShellShock [11] makes it possible to remotely execute commands on machines. According to Delamore and Ko [12] a large group of devices offering services on the Internet are vulnerable for this kind of attack. Not only insecure services can pose a threat, also common secure services can. For instance, a Domain Name Server (DNS) providing domain name to IPv4 address lookup can suffer from misuse. An attackers can use DNS servers to amplify their attack as shown in 2008 by Kambourakis et al. [13].

One possibility for reducing these kind of misuses is based on the ability to detect an ongoing exploitation

and resolve the problem as proposed in [13]. However security breaches are hard to find because no information about the used vulnerability or method is available during the attack offering no indication of ongoing attack making it difficult to note the duration and scale of the breach. A counter measure to reveal security breaches is to create honeypots as described in 2011[14] and 2013 [15]. These are decoy systems meant to be attacked in order to monitor the type of attacks and provide information about the assailant and used vulnerabilities and techniques.

A different approach to the problem of misuses is to prevent possible assailants to set-up an attack, which can be achieved by preventing assailants from gathering the data they need for an attack. The prevention of data gathering is where port scanner detection plays an important role. By detecting ongoing port scans, the scanners can be blocked thereby depriving them of gathering the data which is needed to perform attacks. Considerable research has been done on port scanner detection from publications proposing one method such as done by El-Hakk et al. [16] to complete surveys, comparing different detection methods, as performed by Bhuyan et la. [17]. From the performed research, several classes of scanners and their behaviour can be deducted such as slow and fast scanners, and scanners targeting single, multiple or all ports. However, most research focusses on the detecting of the port scanners but does not further investigate the differences between scanners and their properties.

The analysis of difference between scanners their properties and behaviour can reveal traits which are not used by current scanner detection tools. Behavioural aspects such as cyclic patterns, scanning speed, usage of multiple source addresses and or devices are presented. Also detection and analysis of previously unknown scanning tools is performed. Implementing the traits presented in this thesis into scanner detection tools can result in a higher detection rate of scanners. Therefore this thesis will focus on the classification of scanners, thereby mapping their behaviour into different categories.

The analysis of scanners will be divided in several chapters within this thesis. Chapter 2 contains background information about the protocols used on the Internet. Chapter 3 analyses related work providing an overview of the current knowledge on scanning behaviour and classification. Chapter 4 contains detailed information about the used data set, and the necessary filtering applied to it. Chapter 5 presents an initial analysis of the data in order to get basic understanding of what it's contents are. Chapter 6 focusses on the tools available and used by scanners, as the tools themselves might reveal additional information. Chapter 7 analyses the location of scanners in the IPv4 range in order to determine if certain address ranges are preferred over others. Chapter 8 contains a time analysis is performed, defining behaviour on a time scale. Chapter 9 analyses the different behaviours and traits exhibited by scanners. Chapter 10 presents a discussion followed by a conclusion.

2

BACKGROUND INFORMATION

This chapter is intended to bring a clear view of the Internet and the protocols used to communicate. A basic understanding of the Internet and the used protocols is necessary as the thesis focusses on mapping behavioural traits of scanners scanning the Internet. Most of the traits and behavioural aspects are derived or based on the used protocols as well as the Internet topology.

First, a concise analysis of the studied protocols and communication methods is presented. Followed by an overview of the Internet and its topology.

COMMUNICATION METHODS

This section gives an overview of the studied communication methods. Part of the information has been retrieved from a book written by Andrew Tanenbaum [18] and the appropriate Request for Comments (RFC) [19] provided by the Internet Engineering Task Force (IETF) [20].

THE OPEN SYSTEM INTERCONNECTIONS REFERENCE MODEL (OSI)

The OSI model is developed by the International Standards Organisation (ISO) in order to facilitate communication between systems on the same network, and with other networks. The systems and networks forming Internet use this model in order to connect to each other, thereby allowing interaction with each others services. The OSI model, shown in figure 2.1, consists of seven layers. However in practice, this layer model has been reduced to five layers. In the five layer model, the presentation and session layers (5 and 6) are merged with the transport layer.

The first layer composing the model is the physical layer, as it suggests it is responsible for transmitting bits. The transportation of these bits can be achieved using traditional copper wires but also using fibre cables or other types of transportation. Then there is the data link layer which transforms the stream bits into separate frames. Following the network layer handles routing of the packets from the source to the destination. Next is the transport layer which handles the correct sending and arrival of packets. The session layer allows for users on different devices to establish multiple sessions. The presentation layer is in charge of parsing the data into defined structures which can be used by the application layer.S Finally, the application layer is entirely defined by the application and dictates what the program does with received data or which data it sends.

In this thesis, part of the work is based on the Data Link layer, namely information that can be extracted by examining received frames. Another part of the findings are based on information extracted from packets originating from the Network layer, as they contain meta data. Finally, most of the information presented in the thesis will use information provided by the Transport layer.

THE NETWORK LAYER

As explained earlier, the Network layer describes the routing of the packets sent in a network. The most used protocol on the Internet today is the IPv4 [1]. This is later in the thesis confirmed by an analysis of the collected data, where a majority of the received packets (99.95%) use the IPv4. Therefore it is crucial to understand the IPv4 in order to be able to analyse the received data.

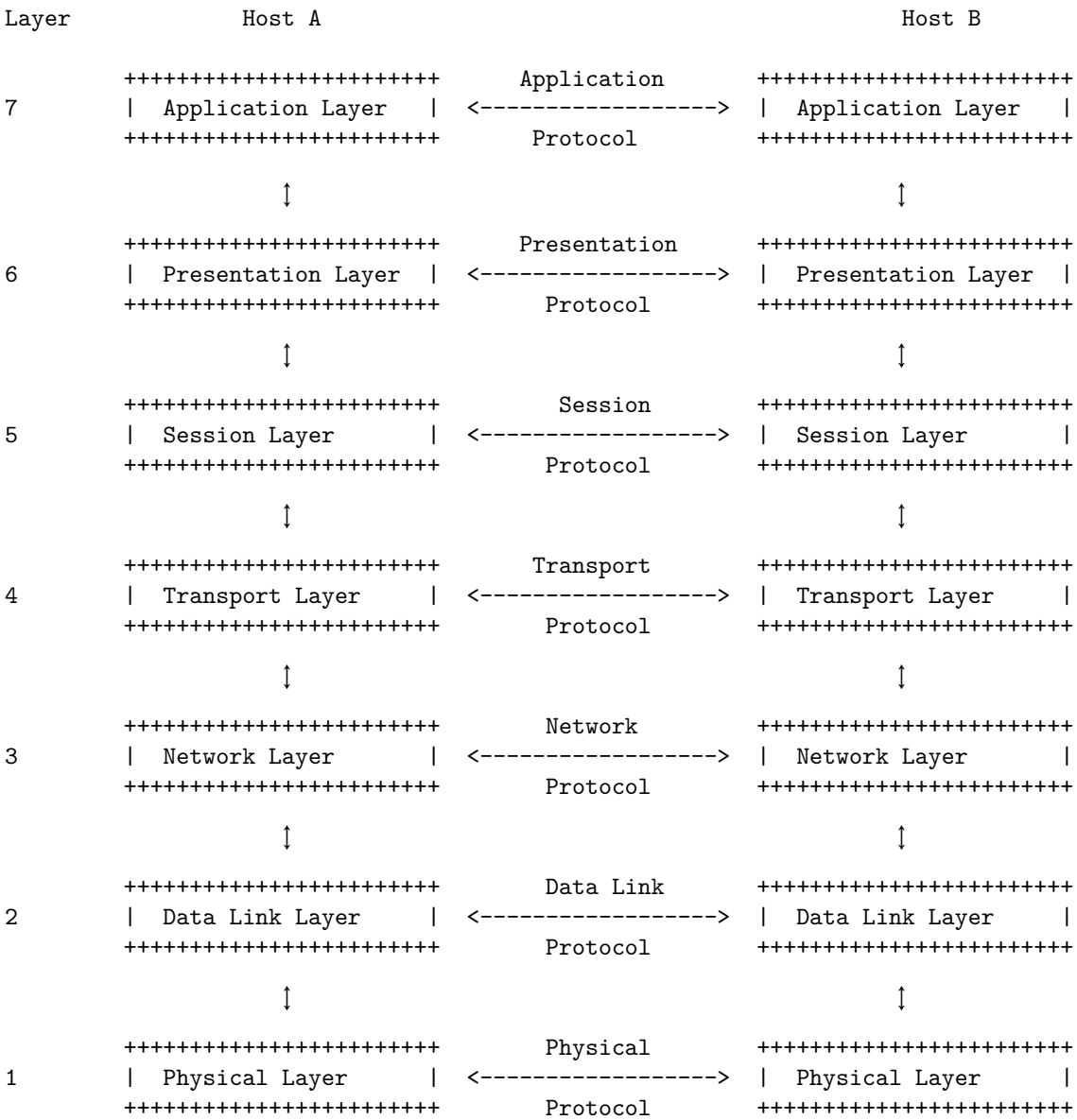


Figure 2.1: The OSI reference model

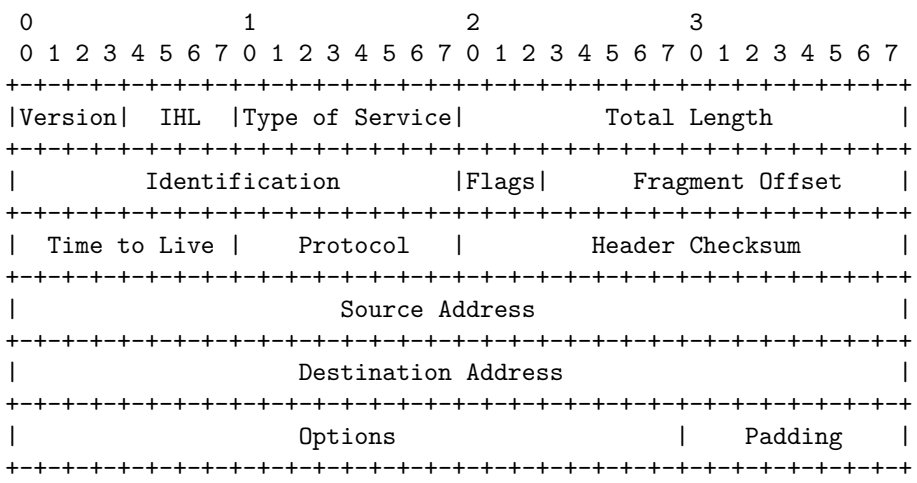


Figure 2.2: IPv4 Header as presented by Jon Postel in the Internet Protocol RFC [1]

1. Decimal	2886794753				
2. Binary	10101100000100001111111000000001				
3. Splitted Binary	10101100	00010000	11111110	00000001	
4. Dot Decimal	172	. 16	. 254	. 1	

Figure 2.3: The IPv4 binary to decimal conversion

In order for the devices to communicate using the protocol in question, they need to store information about the source and the destination of the packets they are sending. This information is stored in the header of the packet, presented in figure 2.2. The numbers on top of the figure indicate the bit and byte number and the boxes delimit the different fields thereby revealing their size. Along with the addresses additional information is stored.

In the header we can see that the destination and source address are stored in fields of 32 bits. These fields are able to represent any IPv4 address, meaning that there are 2^{32} possible addresses in the range (roughly four billion). Additional information is stored in the identification number to ensure proper routing of the packets. The identification number is randomly chosen for packets which no relation with each other, therefore it can be considered to be random in the context of this thesis. The Time To Live value is present in the header and determines how many hops a packets may travel through the network. Each time the packet travels through a device the TTL is decreased by one, when the value is zero the packet is discarded. This allows for time critical packets, or which are in a loop to be discarded thereby not congesting the network. Furthermore some other options and flag fields are present. However, these fields exhibit less significant information in the context of this study and are therefore not further explained. Additional information on these other options and flag fields can be found in the RFC describing the IPv4 [1].

An important aspect, which recurs in this report, of the IPv4 are the addresses. As mentioned before, addresses have a length of 32 bits and can be represented as an integer between 0 and 4,294,967,296. However more commonly, these addresses are represented in a dot-decimal notation. The conversion of the binary IPv4 address to the dot-decimal notation is shown in figure 2.3. This means that the 32 bits are split up into sections of 8 bits called bytes. Bytes are then converted to a decimal format and separated by dots.

The dot-decimal notation offers a way to declare IPv4 ranges which is called the slash notation [1]. It allows to define adjacent addresses as subnets, by adding a slash to a dot-decimal notation followed by a number defining the block size of the subnet. The block size refers to the number of IPv4 addresses available in that subnet. The lower the number in the slash notation the larger the number of IPv4 addresses in the subnet.

In order to calculate the subnet address range, the IPv4 address in the slash notation must be converted back into binary notation. Then the first x bits of that address are kept, where x is the number defined behind the slash. Now the subnet is defined as all IPv4 addresses having the same first x bits as the calculated IPv4

[1] https://www.ripe.net/about-us/press-centre/understanding-ip-addressing

```
1. Slash Notation                 145   .    94    .    11    .    22       /16
2. Convert IPv4 to binary      10010001 . 01011110 . 00001011 . 00010110
3. Keep first 16 bits          10010001 . 01011110 . 00000000 . 00000000
4. Convert IPv4 to dot decimal    145   .    94    .     0    .     0
5. Range                      145.94.0.0 --> 145.94.255.255
```

Figure 2.4: IP range calculation using slash notation

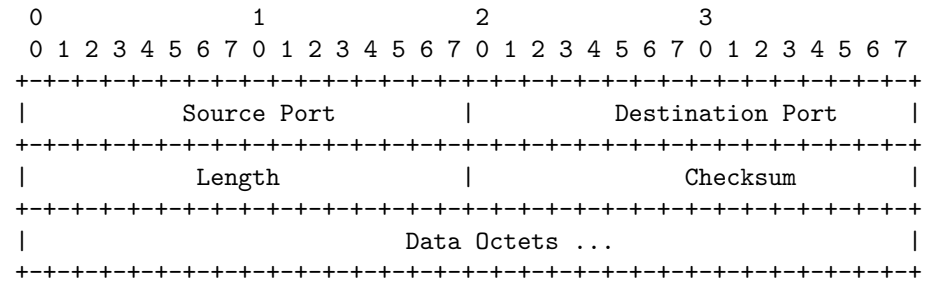


Figure 2.5: UDP Header as presented by J. Postel in RFC 768 [2]

address where the remaining other bits may have any other value (0 or 1). Example using the 145.94.11.22/16 range is given in figure 2.4, illustrating that the defined slash notation range actually covers all addresses from 145.94.0.0 to 145.94.255.255.

THE TRANSPORT LAYER

As explained earlier, the Transport layer is responsible for the correct sending and arrival of packages in a network. The most common protocols are Transmission Control Protocol (TCP) [3] and User Datagram Protocol (UDP) [2]. While both these protocols are used in this report, the main part of the research relies on the TCP. Therefore a more elaborate analysis is given of the TCP, and only a brief explanation of the UDP.

UDP

A significant feature of the UDP is that it is a connectionless protocol. This means that it allows devices attached to the network to communicate without having established a connection. A connection can be explained as a mutual agreement between two devices on the same network to exchange data.

The UDP header is shown in figure 2.5, it only contains a four fields. Only the source and destination ports, the data length and checksum are included. As explained earlier the Network layer assures communication between two machines on the same network. Therefore the addresses of those machines are stored in the IP header, and not in the header of the UDP. The destination ports assure that the UDP packet is delivered to the correct handler of the Interface layer, which forwards it to the application or service associated with that port, allowing different services to run on the same machine. Finally the checksum is used to determine if the packet has been altered during transport by random transmission errors.

The simplicity of UDP, altogether with it being connectionless imply that no error control, timing control or flow control over the packets can be effectuated. The entire UDP specifications are described in RFC 768 [2].

A well known example of the usage of UDP is the Domain Name System (DNS). DNS request can also be performed using TCP, however UDP offer greater speed as no connection needs to be established. It is used to translate the IP address of a server using its name. When browsing through the web a user enters a websites name into a browser. This browser then sends a UDP packet containing the domain name of the website (example: www.google.com) to a known DNS server. The UDP packet is sent for example from port 1234 (Browser) to port 53 (DNS server). The DNS server then responds with a packet containing the IP address of the server hosting the website allowing the user to communicate with it. The response packet is sent from port 53 to port 1234. The usage of UDP for DNS is trivial as no connection needs to be established in order to provide the DNS service.

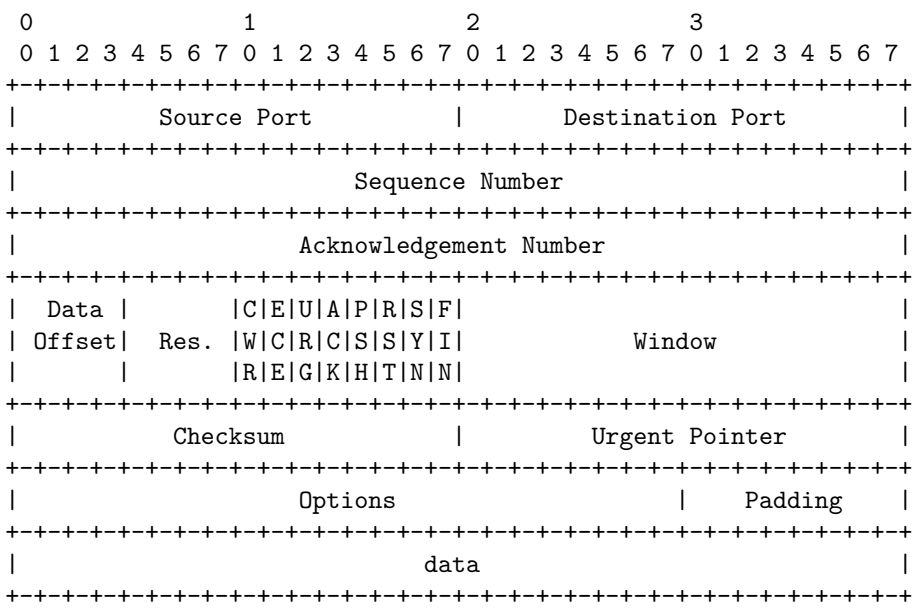


Figure 2.6: TCP Header, defined in RFC 793 [3]

TCP

In contrast to UDP, TCP establishes a connection in order to exchange data. Subsequently, the TCP header contains more information than the UDP header. The full specification of the TCP is defined in RFC 793 [3]. Note that some clarifications and bug fixes have been made to the protocol and are described in other RFCs, especially RFC 3168 [21] which introduces two new TCP flags. Both the connection property and additional data in the header will be used in this thesis.

The TCP header is shown in figure 2.6. As with the UDP header the addresses are not included as they are handled by the Network layer. However as the UDP header, the source and destination ports are defined. These have the same purpose as in the UDP header, specifying to which service on the devices the data should be sent. This allows multiple services to run on the same devices.

Another important field in the header is the sequence number. This number allows the receiver of TCP packets to sort incoming packets into the correct order. The sequence number is randomly selected by the sender at the start of the TCP connection and is incremented when a new packets is sent. As will later be explained in the reports, the randomness of the initial sequence number can be used in the data analysis.

Following the sequence number is the acknowledgement number. This number is used from the sender to receiver to communicate up to which point the packets sent are received, allowing for selective retransmission if packets are lot.

A bit further in the header are eight control bits, also knows as flags, which are located after the reserved field. These fields are used later in the thesis for filtering data, they indicates some properties of the packets.

CWR The Congestion Window Reduced flag indicates whether or not the window size (which is explained next) should be reduced.

ECE The Explicit Congestion Notification flag indicates whether or not there is congestion of the network.

URG The Urgent Pointer flag indicates whether or not the packet should be treated with urgency by the network it is being transported on.

ACK The Acknowledgement flag indicates whether or not the packet is an acknowledgement of a received packet.

PSH The Push Function flag indicates whether or not the packet should be pushed directly to the application, thus not being buffered at the receiver.

RST The Reset flag indicates whether or not the connection between the two parties should be reset.

```
     TCP A                                                TCP B

1.   CLOSED                                               LISTEN

2.   SYN-SENT    --> <SEQ=SA><CTL=SYN>                    --> SYN-RECEIVED

3.   ESTABLISHED <-- <SEQ=SB><ACK=SA+1><CTL=SYN,ACK>     <--  SYN-RECEIVED

4.   ESTABLISHED --> <SEQ=SA+1><ACK=SB+1><CTL=ACK>       --> ESTABLISHED

*5.  ESTABLISHED --> <SEQ=SA+1><ACK=SB+1><CTL=ACK><DATA> --> ESTABLISHED

SA = Random sequence number of TCP A
SB = Random sequence number of TCP B
CTL = Control bits in TCP header
* First packet sent after connection is established
```

Figure 2.7: The TCP three way handshake

SYN The Synchronize flag indicates that the sender wants to create a new connection with the receiver.

FIN This flag indicates that the connection should be closed.

Next to the control bits is the window size field which contributes to the flow control of the TCP. It dictates how many packets can be in transit between the sender and receiver. There is no fixed value for this field and is often dictated by the used operating system or device hardware.

The checksum is used for error checking in the data.

The other fields present in the header are used for miscellaneous information which is not in the scope of this thesis. For additional information about the usage of these fields please refer to [18].

As mentioned in the beginning of this section, TCP is a connection oriented protocol. Meaning that the devices communicating, using TCP, establish a connection. This is done using a three way handshake depicted in figure 2.7.

In order to establish the connection, two parties are needed, in the figure called *TCP A* and *TCP B*. First there is the Initiator (*TCP A*), a device which desires to establish a connection. The Listener (*TCP B*) is a device offering a service, and therefore listening to incoming TCP packets on a certain port. To initialize the connection, the initiator sends a TCP synchronization packet to the listener. This synchronization packet has only the SYN flag set, and a random sequence number SA. When the Listener receives this synchronization packet it acknowledges its receipt to the Initiator by sending a synchronization acknowledgement packet. This packet has both the SYN and ACK flags set where the sequence number SB is randomly selected but the acknowledgement number is SA incremented by one. Finally the Initiator receives the synchronization acknowledgement packet of the Listener and acknowledges it. The Initiator does this by sending an TCP acknowledgement packet (an empty TCP packet with the ACK flag set). As we will later be shown in the report, this three way handshake is key to the ability to scan the Internet for devices offering services using TCP.

TOPOLOGY OF THE INTERNET

The Internet as mentioned before is a network of networks of devices communicating with each other using different protocols on different layers as defined by the OSI model. The most commonly used protocol is the IPv4 which is used to assign addresses to devices in order to route traffic from one machine to another. The number of addresses is finite as they are represented by a fixed number of bits.

The assignment of the addresses to organizational units is performed by the Internet Assigned Numbers Authority (IANA) operating on a world wide scale. IANA, assigns blocks of addresses to Regional Internet Registries (RIR), which operate on a continental or country level. Regional operators then redistribute the assigned blocks to Local Internet Registries (LIR) which are given by Internet Services Providers, large organizations, companies or research institutes. Finally, providers assign their blocks to customers.

However, routing all traffic based on the IPv4 addresses would imply that each hub in the network is aware of all IPv4 addresses in the world. Therefore, IANA allocates Autonomous System (AS) numbers associated with IPv4 blocks possessed by RIRs. IPv4 blocks are associated with AS numbers allowing hubs in the network to exchange only the AS numbers to inform other hubs about their connectivity, reducing the amount of information to be exchanged between them. This is done using the Border Gateway Protocol [22]. In addition, RIRs give AS numbers to LIRs meeting the requirements (often related to the amount of addresses allocated to them), further reducing the communication needed between hubs.

3

RELATED WORK

As mentioned in the introduction, the majority of the research on port scanning focusses on the detection of ongoing scans. Studies on scanner detection can offer insights into the behaviour of the scanners, as in order to detect scanners, properties defining them must be known. Several techniques are used for the detection of ongoing scans such as rule based approaches and network flow analysis. The used detection methods allow to derive different scanner properties such as fast and slow scanners or differences in number of ports or addresses probed.

Rule based approaches look at incoming traffic and determine if there are ongoing scans by employing simple rules which can be translated into conditional clauses as presented by Kanlayasiri et al. [23]. The authors research the detection of TCP port scanning activity, distinguishing between several sorts of probing techniques. Probing techniques are related to the flags set in the TCP header of the scan probe, and can be classified into three classes. The classes consist of connection, stealth and syn probing, which are explained in detail in subsection 6.1.3. Differentiation of header values is an indication that the behaviour of scanner is not uniform as they use distinct probing techniques.

Besides from differentiating between flag values in the header of TCP probes, the authors propose a rule based algorithm to detect scanners. The rule based algorithm looks for incoming packets having the specified TCP flag values set. If the flag values correspond to a value in the F tuple $F = \{URG, PSH, SYN, FIN, ACK, RST\}$, then the packet is marked as a probe with associated flag and processed. The number of probes are stored for a predefined amount of time for each source address. When the number of unique destination ports of the recorded probes with associated flag exceeds a threshold, the source address is marked as a scanner. The used rules in the algorithm suggest that scanners target multiple destination ports within a defined amount of time, as the detection is based on number of different ports scanned given an amount of time.

From [23] three properties of scanners can be extracted, scanners use different probing techniques scanning for different ports, and do so in a certain time interval. The authors present results showing that scans using different probing techniques are detectable. Using the rule based algorithm, the authors are able to detect different used scanning tools . The results confirm that the three defined properties are indeed enough to detect scanners, thus that these properties are exhibited by scanners.

Fuzzy logic approaches are an extension of rule based methods by being more tolerant in terms of rule input and output. Dickerson et al. [24] propose a scanner detection system based on fuzzy logic. As in [23], the authors also make a distinction between sent probes. Three different scanning probes are defined based on the protocols used, namely TCP, UDP and ICMP probes. The distinction between the three protocols shows that scanners probe ports responding to different protocols (UDP and TCP), and scan for devices connected to Internet without specifying port number (ICMP). The proposed algorithm by [24] does not use hard set thresholds for the number of destination addresses and ports probed by the same source address. The use of variable thresholds reveals that scanners have different behaviour in terms of number of destination addresses and number of destination port scanned.

Bridges and Vaughn [25] researched the efficiency of both rule based, and fuzzy detection systems. In the paper the authors reveal that detection systems based upon fuzzy logic have a higher detection rate than rule based detection systems. Implying that the detection of scanners is more efficient when applying rules which

allow for variable ranges of input and output. The higher detection rate of fuzzy rule based detection systems indicates that scanners exhibit variance in the traits used by the algorithms to detect and attacker. Entailing that not all traits displayed by scanners are identical but that scanners do exhibit similar behaviour as the behaviour is determined by the rules used in the algorithm.

Network flow analysis is another technique employed to detect port scanners. First a ground truth is created by analysing network statistics, of a clean network (not under attack), such as the number of packets routed to each destination, creating a flow map. Then the flow is constantly monitored and compared to the ground truth, allowing the detection of anomalies. Ertoz et al. [26] use network flow analysis to detect anomalies generated by scanners sending probes to networks. The authors use several criteria to rate the collected flows, such as number of connection classified by source and destination ports. The flows are compared based on these criteria and outlier are considered to be generated by scanner. Therefore, one may conclude that scanners indeed have a different behaviour compared to regular traffic thus making it possible to classify them.

Ertoz et al. [26] add a new distinction between scanners which in the previous two researches [24], [23] is not made. The authors differentiate between normal scanners and slow scanners, sending one probe per minute or even per hour. The distinction between the two scanning speeds is reflected in the proposed detection system, which in the case of the slow scanners does not take time into account.

The notion of different classes of scanners is confirmed by Dabbagh et al. [27] where the authors focus on the detection of slow scanners. According to the authors, slow scanners try to remain undetected by lowering the number of probes sent to a targeted network to evade most rule and fuzzy based detection systems, or time based in general. Because these systems store potential scanners for a limited amount of time. Thus by lowering the scan speed the detection system will not raise an alarm because the scanners did not target the network frequently enough.

All presented papers acknowledge different types of scanners such as vertical scanners, targeting all ports on one address. Horizontal scanners targeting a single port on all addresses, or strobe scanners targeting multiple ports on multiple addresses. However no overview of different types of scanners and their traits is given. A survey written by Bhuyan et al. [17] on port scanning detection does offer an overview of types of scanners. Furthermore the notion of distributed port scans is described, where multiple source IP addresses are employed to scan a network.

The classification presented in the survey is based upon work which primarily focusses on detecting scanners and not classifying them. Therefore the identified classes are not deduced from data analysis but out of the methods and rules used to detect ongoing scans. The classification of scanners through their detection methods may result in undiscovered classes, as no research for detecting undiscovered classes based on data analysis is performed. Also no information is given about the differences between scanners in the defined classes. In the horizontal scanner class, there might be some differences between the scanners, as one could argue that two scanners can have different speeds, due to their network connection or settings. Mapping these subtle differences, along with discovering undefined classes are not the goals of previous research.

As mentioned in the beginning of this section, research performed on data resulting from scanning activity is sparse. In a publication from 2003 [28] a data driven approach to classifying scanners is adopted. A distinction is made between used protocols, and the type of TCP packets used to perform the scans. Definition of different scanners is given, such as horizontal and vertical scanners, which is complemented by the distribution of scanned targets per scanner. Lee et al. [28] provide more information about differences between scanners belonging to the same class such as horizontal scanners by distinguishing them by number of targets probed. More interesting is the representation of scanners in terms of their location on the world. Even though this research adds some interesting insight into the classification of scanners and their behaviour within those classes, there is one shortcomings. Although the representation of scanners on a world map is interesting, it does not correct for the population in an area or the number of available Internet connections.

Research performed by Durumeric et al. [29] is also data driven. In their paper, a clear distinction is made between used tools or programs to execute the scan. Fingerprints are presented for ZMap and Masscan, this allows for classification based on the tools used by the scanner. The authors combined the used tools with the scanned destination ports which reveals that packets generated by ZMap are sent more often to port 443 than packets generated by other tools combined. Another distinction made by the authors is the difference

between large and small scans, the former sending a high amount of probes whereas the latter only sends a few. Also the authors analysed the geological location of the scanners, similar to the work presented by Lee et al. [28]. In addition they include an analysis of providers used by the scanners were sending packets.

The article written by Durumeric et al. [29] provides new classification of scanners based upon the used tools, and insight about the behaviour of scanners in those classes. However the authors filtered data from scanners known as slow scanners, sending a low amount of packets over an extended period of time, potentially excluding slow scanners from the results. As in [28], the numbers given on the geological location of scanners do not provide a relative view, in terms of population and internet connections, on their global distribution.

From the related work it becomes clear that some kind of classification can and has been done regarding scanners. Research focussing on the detection of scanners presents basic classifications of scanners such as vertical and horizontal scanners, Furthermore elaborate class definitions are given from studies which have a data driven approach. The classes defined in data driven research distinguish scanners by used tools, scanning speeds, and geographical location. Therefore the related work strongly indicates that it is possible to differentiate between scanners other than based on ports and destination addresses scanned, thereby enabling detailed a classification. Raising the question whether there are other characteristics exhibited by scanners not covered by related work. In addition, the presented related work provides results which can be used to validate the results presented in this thesis such as ratios of used protocols. Finally the related work reveals some challenges which should be addressed while performing research and analysis such as data sanitation. In the next section a closer look at the data set used in this thesis is presented, in order to anticipate on possible results being biassed.

4

DATA COLLECTION

This chapter provides an introduction to the used data during the research and explains where and how the data has been collected. Also, clarifications are given on why parts of the data are excluded from the research, the main argument being that the data is polluted not offering a reliable source to perform analysis on scanning behaviour. In addition challenges are included which where encountered during the handling of the data.

DATA PROVENANCE

The used data in this research is provided by the TU Delft. The data set is based on is unroutable traffic received by the university during the month of April 2015. This means that packets received by the TU Delft are collected, for which the destination IP address does not exist within the network. More specifically, the destination address of packets received, lie within the network address range of the university but no device is associated with it. Normally this data is discarded however for research purposes the packets are stored on a server for further analysis.

The data is captured on a server of the TU Delft which routes all the incoming traffic of the university thus ensuring that all unroutable traffic is captured.

The capturing of the data is done using Tshark [30], a network traffic capturing tool. The data is stored in its raw form using the libpcap file format [31]. The total amount of data captured in the month of April 2015 amounts to 366GB. Storing the unroutable packets in their raw forms implies that packets are stored bit for bit without any alteration to the entire content of the packet. This ensures that the stored data is a true representation of the received unroutable traffic. More so, the libpcap file format allows to add time stamps to the each received packet, providing additional meta data.

As mentioned, the data originates from unroutable traffic sent to the TU Delft. More specifically the data originates from the three IPv4 blocks assigned to the university:

- 130.161.0.0/16, private network
- 131.180.0.0/16, private network
- 145.94.0.0/16, public network

Each block has 65,025 addresses, which amounts to a total of 195,075 IP addresses. The private network ranges are used by the employees and network equipment of the TU Delft, whereas the public range is used by students and visitors. Both networks are reachable from the Internet. Specifically, the public network offered by the TU Delft is part of the education roaming world-wide roaming access service named Eduraom [32] allowing any visitor, which is member of an organisation associated with Eduroam, access to the network.

BASIC ANALYSIS

In order to get a better impression of the collected data some basic analysis is described in this section.

Subnet	130.161.0.0/16	131.180.0.0/16	145.94.0.0/16
ICMP	20,022,005 (3.02%)	4,929,675 (2.74%)	28,912,013 (1.22%)
TCP	498,943,808 (75.5%)	138,283,932 (76.92%)	509,846,048 (21.67%)
UDP	141,854,168(21.46%)	36,438,778 (20.28%)	1,813,536,308 (77.09%)
IPv6E	544 (0.00%)	23,782 (0.01%)	157,887 (0.00%)
Other	132 (0.00%)	34 (0.00%)	544 (0.00%)
Total packages received	660,820,657	179,676,201	2,352,452,800

Table 4.2: Number of packages received from each transport layer protocol type (percentage)

The first metric used to assess the data is the ratio of used protocols in the network layer as defined in the OSI model. This is achieved by parsing all the captured frames on the data link layer and inspecting their ethernet header. The Ethernet header contains the used ether type which corresponds to the Network layer protocol. The results are presented in table 4.1.

CGMP	89,736 (0.00%)
STP	1,341,030 (0.04%)
IPv4	3,195,362,408 (99.95%)
ARP	49,731 (0.00%)
Total frames received	3,196,842,905

Table 4.1: Number of frames received from each network protocol type (percentage)

There are four network protocols used in the collected data. First is the Cisco Group Management Protocol (CGMP) which is a proprietary protocol from Cisco, used by network equipment, so as routers and switches, to establish multicast group memberships [33]. Second is the Spanning Tree Protocol (STP), which is used by network equipment to ensure that the network does not contain loops. Third is the Internet Protocol version 4 (IPv4) [1], explained earlier. Finally, the Address Resolution Protocol (ARP) [34], used to report network addresses.

It is clear from the results presented in table 4.1 that not all network protocols have the same share of received traffic. The majority of the captured traffic, 99.95%, uses IPv4. Therefore a further look into the packets sent using the IPv4 protocol is desired. As the data provided by the TU Delft is collected from three different subnets, the analysis will be performed on these subnets separately. The main motivator for the individual analysis of the subnets is their nature, two are used by employees only whereas one can be used by any person having a Eduroam account. This allows for comparison between them and can reveal potential differences introduced by the network users and therefore usage. The differences, found in the constitution of datasets will be explained and further steps are taken to limit possible bias of the research as much as possible.

Table 4.2 presents the number of IPv4 packets received for each subnet classified by their transport protocol type. The table shows that the majority of the sent packets belong to four different protocols. First, the Internet Control Message Protocol (ICMP), which is mainly used to report error messages [35]. Second is the Transmission Control Protocol (TCP), which is a connection oriented communication protocol [3]. Third is the User Datagram Protocol (UDP), which is a connectionless communication protocol [2]. Followed by the the Internet Protocol version 6 Encapsulation (IPv6E), which encapsulates IPv6 packets into IPv4 packets in order to send them through a network [36]. Finally, all other packets using different protocols are added in the other protocols (such as Gateway-to-Gateway, Internet Group Management, and Inter-Domain Policy Routing Protocol).

The results in table 4.2 show that the data captured from the public and private subnet differ. The ratio of TCP and UDP packets in both networks are inverted. For the private networks the TCP and UDP traffic represent respectively around 75% and 20%, whereas for the public network the TCP and UDP traffic represent respectively 20% and 75% of the traffic. Additionally the percentage of ICMP packets of the public network is two times smaller than the percentage of ICMP packets received on the private network. The different ratios of protocols in the private and public networks will be explained in the data sanitation section of this chapter.

Furthermore table 4.1 shows a discrepancy in the presented results. According to the results presented in table 4.2 there are in total 3,195,362,408 packets using IPv4. However, adding up the number of packets using the IPv4 from all three subnets results in 3,192,949,658, suggesting that 2,412,750 are unaccounted for in the analysis of the subnets. The unaccounted packets all have destination addresses outside of the ranges of the TU Delft explaining why they are not present in the analysis of the three subnets. The presence of these packets in the general dataset is due to an expired Time To Live parameter which renders them unroutable.

In the next section a sanitation of the data is proposed based upon results shown in this section.

DATA SANITATION

As mentioned before, the data consists of unroutable data of three subnets of the university. However, not all data can be analysed as the combination of all different protocols against different analysis metrics reaches beyond the scope of this thesis. Therefore some filtering needs to be applied. In this section the data sanitation process is explained, and arguments are given to explain reasons of the employed filters.

NETWORK LAYER SANITATION

As shown in the previous section, the majority of frames, sent to the TU Delft is of the IPv4 type (99.95%). The other protocols are CGMP, ARP, and STP. These three protocols will not be further analysed in this thesis. They are commonly used by the infrastructure connecting the devices creating the network. Also, a manual lookup in the data using these protocols suggests that this traffic is solely generated by devices within the TU Delft, rendering it useless for this thesis.

As mentioned in previous section, approximately 2.5 million packets are not considered in the analysis, because the reception of packets outside of the IPv4 address range of the TU Delft by the router capturing the unroutable traffic. A manual inspection of the data confirms this observation, and shows that the majority of these packets are ICMP packets using the IPv4. These ICMP packets all exceeded their Time To Live parameter and are therefore seen as unroutable.

Analysis performed in a later stage of the research acknowledges that data should further be sanitized. The data contains packets which are sent from the IPv4 range of the TU Delft. Manual analysis reveals that the unroutable traffic generated from within the TU Delft is directed to a single destination address. The single destination of the unroutable traffic suggest a misconfiguration (wrong destination address) or a server being unavailable (causing packets sent to it being unroutable).

Packets using the CGMP, ARP, and STP are ignored during the rest of the research, resulting in a dataset containing only IPv4 packets. Furthermore, only packets directed to the three subnets of the network of the TU Delft will be analysed, and packets originating from addresses within the TU Delft will be ignored.

TRANSPORT LAYER SANITATION

This section investigates whether sanitation at the Transport layer is necessary. When looking at the percentages of packets received per transport type protocol, one can clearly see that the TCP and UDP represent the majority of the traffic (approximatively 97.00%) for both the private and public networks of the TU Delft. Therefore, due to the low share in traffic the IPv6E and other protocols will not be considered in this thesis.

Another protocol, consisting of approximately 3% of the network traffic, is ICMP an error propagation protocol. Even though ICMP can be used for scanning, this research does not focus on the errors of the studied network. Therefore packets using this protocol will not be part of the analysis of the data.

SUBNET SANITATION

Now that the data has been sanitized of some network and transport layer protocols, the three different subnets of the TU Delft should be analysed. When looking at the different ratios of UDP and TCP packets received on the private and public network there is a complete shift. This confirms the suspicion raised in the previous section that the private and private networks receive different unroutable traffic.

Although analysing both private and public parts of the TU Delft subnets is feasible, ignoring the public part seems the better option. Filtering out the public part may be justified by its wireless property. The wireless property of the public network allows users to abruptly close a connection. This can be done by turning off a cell phone or closing a laptop. This can cause the other device which connected with the laptop or cell phone to keep on sending packets, which have become unroutable. A service using a TCP connection will stop sending packets once the window size is full, and will close the connection when a time out occurs when the receiving party does not acknowledge. However, a service using a UDP connection will keep on sending packets, as no feedback from the receiving party is required. This is one possible explanation of the difference in the UDP and TCP ratios of the public and private network. However, it is not proven and remains a hypotheses therefore similar research should be consulted to investigate what ratios are usual for

unroutable traffic.

When looking at similar research performed by Wustrow et al. [37] the same ratios of TCP and UDP unroutable packets are found as in the private networks. The ratios are also 70% and 30% for respectively TCP and UDP packets. The research has been performed on packets originating from unroutable traffic of unused portions in 35 /8 IPv4 address blocks. The collection periods lasted for one week each during a time period of five years. The authors verified the results by collecting data for three separate days yearly confirming the observed TCP and UDP ratios. Therefore the results given in the research can be seen as representative for the normal ratios between UDP and TCP for unrouteable traffic.

To be sure that the analysed data is clean of possible unwanted traffic caused by closed connections, and to match the TCP and UDP ratios presented in other research, the data of the public part of the TU Delft network will not be used in the rest of this report.

The data sanitation process yields data which consists of unroutable traffic sent to the private range of the TU Delft, and not originating from the university. In addition the used dataset only contains TCP and UDP traffic which is captured during the entire month of April 2015, resulting in 815,520,686 packets.

CHALLENGES

During the analysis performed in the initial stage of the research, challenges emerged due to the volume of the data.

Initially, Wireshark [38] was used to perform an analysis of individual files. From this initial analysis a feel for data could be derived, such as visualisation of the packets and the ratios of used protocols. However, when handling multiple large (> 200MB) files it quickly became apparent that the memory consumed by Wireshark would form a bottleneck. Opening five files with a combined size of 2GB caused a memory consumption of over 12 GB when performing several queries such as filtering IPv4 addresses and transport layer protocols Also the time taken to perform simple queries felt disproportional to the work performed. These problems are known in the Wireshark community and can be alleviated by changing some display options. However, the problem of memory consumption persists when multiple files are opened or more queries are performed.

Therefore an alternative solution is necessary to perform the file analysis. One option is to use TCPDump to perform the file analysis. This program is console based which greatly reduces the memory footprint. However, TCPDump does not provide custom output files. Also, the ruled base approach to perform analysis can be cumbersome to use if even possible at all.

Accordingly the decision is made to write analysis software based on the libpcap library. It is the library used by TCPDump and Wireshark on Linux/Unix environments as well. The software is written in C++ for performance reasons, the ability to perform bitwise operations, and the compatibility with the header files used for networking by the kernel; which are all needed to perform analysis on a large dataset containing TCP and UDP packets.

The custom analysis software greatly improved the parsing speed and memory footprint. However, due to the size of the collected data, the memory footprint still prevails. Initially the data was converted from libpcap format to a custom binary format discarding part of the information stored in the Network and Transport layer headers, reducing the memory footprint.

The custom data format was later dropped as the discarded information could be used for analysis of the data. Hence the libpacap file format was reinstated, but processing of data was performed by a server environment, having 250GB of RAM allowing the analysis of the data using the custom software without the concern of memory footprint. An additional benefit offered by the server was the fast storage which further decreased the processing time.

5

DATA ANALYSIS BASED ON USED PROTOCOL

This chapter will focus on the initial data analysis which consists of information gathering about the data from a packet's perspective. Destination addresses and ports, along with header values of the different protocols used are studied. As explained in the previous section, the data has been sanitized and only consists of the IPv4 traffic received on the private networks of the TU Delft. In the first part an overview of the targeted address space will be given. The two following sections will be dedicated to the packets using the UDP and TCP, including the values of the header values in these probes.

TARGETED DESTINATION ADDRESSES

An interesting property of the unroutable data is whether or not the IPv4 addresses owned by the TU Delft are scanned with the same frequency. If this is not the case, it could indicate that certain parts of a network are scanned more rigorously for some particular reasons.

The two private networks of the university consist of a total of 130,050 IPv4 addresses. From those addresses only 60,025 where not assigned to devices on the network and thus received unroutable traffic. The total number of packets sent to the unallocated addresses is 827,097,686. Figure 5.1 shows the cumulative distributions of received packets per unallocated address per protocol.

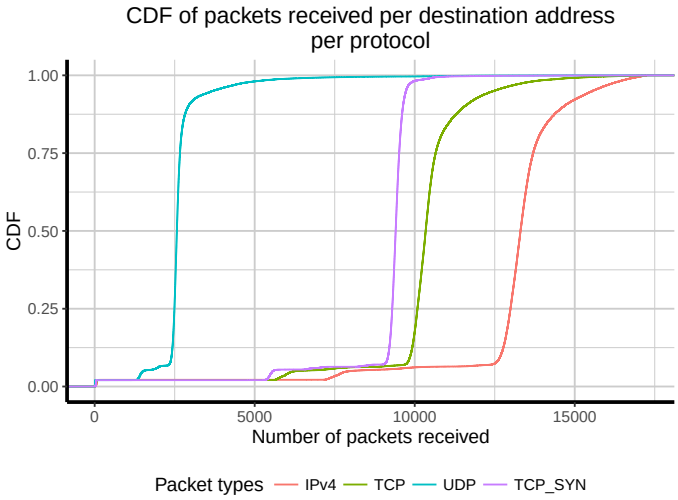


Figure 5.1: Distribution of number of packets received per IPv4 destination address

The average number of received IPv4 packets per destination address is 13,779, and the standard deviation is 30,602 packets, 2.2 times the average number of packet received. The high standard deviation suggests a significant difference in the number of packets received per destination address, implying that some addresses are targeted more than others. However figure 5.1 showing the cumulative distribution received IPv4 packets per destination address (in red), indicating that the majority of the destination addresses received the same amount of packets, roughly 13.300. Exact calculations reveals that 81.75% of the destination addresses receive on average between 12,500 and 14,000 packets.

Some exceptions can be noticed when observing the distribution of targeted IPv4 addresses denoted by the red curve in figure 5.1. Two additional groups of addresses can be seen receiving respectively around 7,500 and less than 100 packets. For completeness, Appendix A shows in figure A.1a the packets per address instead

of the cumulative distribution. From figure A.1a becomes clear that three groups of addresses (in blue) received significantly less IPv4 packets, less than 100 per address. Also another larger group of addresses, having received slightly less packets than the average, can be observed in figure A.1a.

The manifestation of groups receiving less packets can be explained by the allocation of these IPv4 addresses because the addresses receiving less traffic all belong to the same subnet. It is likely that during the data collection period these blocks have been allocated to devices connected to the network, thereby making the traffic destined to those addresses routable again thereby stopping the data collection. Also it could be that some blocks during the time period have been temporary unreachable due to maintenance or malfunctioning.

In contradiction, some IPv4 addresses received notably more traffic than the average, figure A.1a in Appendix A shows two such addresses in the form of two red dots. Further manual inspection shows that the majority of the traffic towards destination addresses, receiving an unusual high amount of packets, is generated by a small amount of source addresses. Such observation, leads to the assumption of an existing misconfiguration or even possible infection of malicious software of the sending party.

The analysis of the number of packets received per destination address reveals that the majority, approximately 81.75%, receive the same amount of packets. Two groups of addresses are noted receiving significantly more (order of million) or less (order of hundred) packets than the average of 13,779. The group receiving less packets is explained by the allocation of IPv4 addresses, the cause of other addresses receiving more packets will be justified in later analysis examining UDP and TCP protocols. The cause of both groups receiving an abnormal amount of packets is validated in favour of concluding that each address receives the same amount of IPv4 packets. Thus each destination address in the monitored subnets is targeted equally in terms of IPv4 packets.

UDP

The data captured using the UDP protocol consists of 166,734,982 packets. The UDP header of the packets carry information namely the source and destination port of the received unroutable traffic. In order to get a better insight of the received UDP traffic an analysis is performed not only on the targeted IPv4 space but also on the destination and source ports.

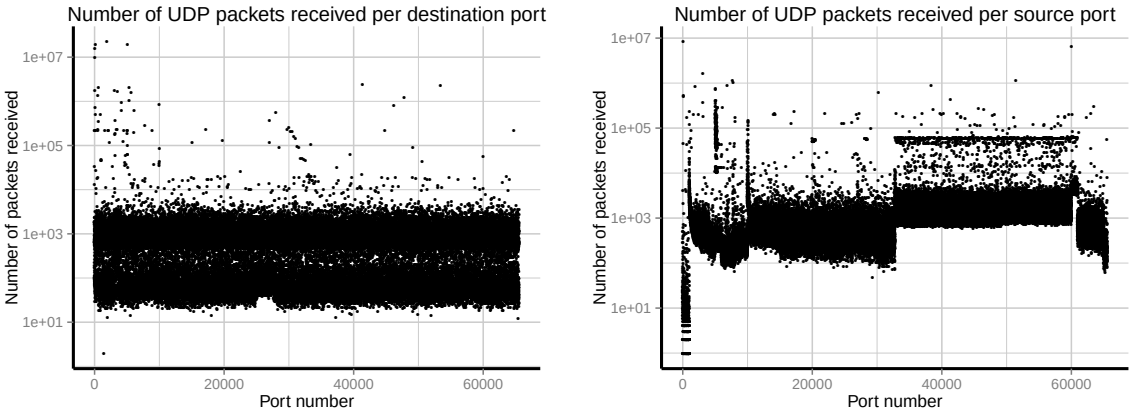
TARGETED DESTINATION ADDRESSES

As mentioned in the previous section, the targeted destination addresses may reveal information about the scanners and the used dataset.

The cumulative distribution of received UDP packets in figure 5.1 (blue curve) shows that the majority of destination addresses receive 2,777 packets. The standard deviation of the number of received UDP packets is 5374, two time the average number of packets received, suggesting that addresses received different amount of UDP traffic. As with the number of IPv4 packets received, the hight standard deviation suggest that the number of packets received per destination address varies. However the CDF curve, shows that the majority 85% received between 2,400 and 2,600 packets, indicating that 85% of the destination addresses received the same amount of probes.

Similar to the number of IPv4 packets received per destination address, some addresses receive less UDP packets represented by two peaks in figure 5.1at around 1,600 and less than 100 packets received. Addresses receiving less packets than the average is confirmed by figure A.1b in Appendix A, which is identical to figure A.1a showing the number of IPv4 packets received per destination address. The similarity in both figure A.1b and A.1a allows for the same conclusion. Addresses receiving less UDP traffic (nealry 15%) belong to the same subnets, which during the collection period were allocated or temporarily unreachable.

The similarity of both figure A.1b and A.1a, together with a manual inspection reveals that destination addresses of the two subnets of the university are targeted equally in terms of UDP packets. Therefore, as with the IPv4 packets, we can conclude that the unroutable UDP traffic does not target specific ranges from the network.



(a) Number of UDP packets received by destination port (b) Number UDP of packets received by source port

Figure 5.2: Number of UDP packets received per port

DESTINATION PORTS

The destination ports of the received UDP packets can give insights into whether or not some services are targeted more often than others. Figure 5.2a shows the number of packets received per destination port. From the figure it becomes clear that there are some destination ports which receive considerably more packets. Some ports receive millions of packets while on average a destination port only receives thousands.

As shown by Table 5.1, 58% of the UDP traffic is directed to ten ports. A preference in ports scanned is expected as from the 65,535 available ports, only port 0 to 1,024 [39] are specified by the Internet Assigned Numbers Authority (IANA) [40] to be used by systems services. However, port numbers higher than 1,024 are used by known services such as port 1,900 which is used by plug and play devices to receive messages from other plug and play devices. There are however two exceptions in the top 10 ports scanned. Ports 41,346 and 53,413 are both unassigned by the IANA. Unfortunately no information can be found on port 41,346 other than it being used by online games. Nevertheless port 53,413 can be associated with security problems. As stated by Trend Micro [41], some routers sold on the market have a backdoor listening to port 53,413 allowing attackers access to the device.

Port	*Service	Number of packets
1,900	Simple Service Discovery Protocol	22,118,339 (13.26%)
5,060	Session Initiation Protocol	19,715,197 (11.82%)
123	Network time Protocol	19,063,163 (11.43%)
53	Domain Name Service	16,038,082 (9.61%)
19	Character Generator	9,881,390 (5.92%)
41,346	Unassigned	2,447,655 (1.46%)
53,413	**Unassigned	2,338,075 (1.40%)
623	***Unassigned	2,102,023 (1.26%)
5,351	NAT Port Mapping Protocol	2,007,395 (1.20%)
3,076	Orbix 2000 Config	1,714,463 (1.02%)
Total	-	97,425,782 (58.43%)

Table 5.1: Top 10 destination ports, in terms of received UDP packets (percentage)
*Service specified by IANA [4]
**Netis System Backdoor
***ASF Remote Management and Control Protocol

The known associated ports can be linked to vulnerabilities. Port 1,900 is scanned for possible Universal Plug and Play (UPnP) vulnerabilities as explained by Beedgen [42]. The UPnP protocol can be misused in three different ways to perform DoS attack. Port 5,060 is the standard port used by Session Initiation Protocol (SIP) which can be actively misused for several kind of purposes as explained by Geneiatakis et al. [43]. The SIP is vulnerable for 12 different attacks leading to possible DoS or unauthorized access. Port 19 can be misused to perform DoS attacks as demonstrated in 2004 by Specht and Lee [44]. All ports presented in Table 5.1 are associated with known security risks, except for port 41,346. This gives a clear indication for the motives of the scans, which is to find vulnerable services.

It is clear that the targeted ports are not uniformly distributed and that some ports are scanned more

intensively, 58% of the scan probes are sent to 10 of the 65,535 ports. The nature of the ports and associated services strongly indicate that the intention of the scanners is to unveil vulnerable services. Another interesting aspect is to see whether some source ports are preferred over others.

SOURCE PORTS

The source ports from which the unroutable UDP traffic originates may expose additional information about the scanners. Figure 5.2b shows the number of packets captured coming from a specific source ports. The plot shows an interesting distribution of the number of packets sent from the different ports. Specifically a clear distinction between two port ranges in terms of received packets. It is apparent from the figure that some ports and even port ranges are preferred over others. Table 5.2 shows the top 5 source ports which are responsible for 11.18% of the traffic.

Source port number	Number of packets
53	8 222 740 (4.93%)
60000	6 454 751 (3.08%)
3076	1 671 169 (1.00%)
7678	1 160 980 (0.69%)
51413	1 138 974 (0.68%)
Total	18648614 (11.18%)

Table 5.2: Top 5 source ports, in terms of sending UDP packets (percentage)

Most of the used ports to send traffic are in the user or dynamic range between 1,024 and 65,535. These ports are responsible for 94.13% of the generated traffic. However as Table 5.2 shows, some of the used ports are within the system range. Notably port 53, used for DNS, generates 4.93% of unroutable UDP traffic. In total, 5.17% of the UDP packets received is sent from system reserved port. This reveals that a substantial amount (95,35%) of the received UDP packets from system reserved ports originate from port 53.

The use of system reserved ports, especially port 53, and the abnormal distribution of used ports shows different scanning behaviour. Some ranges port 30,000 till 60,000 are more frequently used whereas in general ports below 1,024 are significantly less used. However the use of port 53 is remarkable, research shows that the unroutable traffic originating from system reserved ports can be attributed to backscatter.

TCP

As UDP, the TCP header provides additional information compared to the IPv4 header. Not only are the source and destination ports present, but also fields necessary for establishing and maintaining the connection. Therefore a more elaborate analysis must be done on the TCP packets received.

As with the UDP, first an analysis of the scanned destination addresses will be done in order to see if certain subnet ranges are preferred over others. Then the destination and source ports of the TCP packets are evaluated. Finally the other fields of the TCP header are investigated to see whether there are also common patterns.

TARGETED DESTINATION ADDRESSES (TCP)

The targeted address space of the TCP packets can reveal information whether or not specific parts of the network are targeted more often than others. Figure 5.1 shows that the targeted address space (green curve) is almost identical to the targeted IPv4 address space (red curve). The IPv4 and TCP curves differ in the average number of packets received.

Flag value	Number of packets
SYN	558,804,029 (87.91%)
SYN + ACK	62,603,309 (9.84%)
RST	4,999,955(0.78%)
Total	626,407,293 (98.54%)

Table 5.3: Top 3 set flags in terms of received TCP packets (percentage)

Figure A.1c in Appendix A confirms that the same subnets which received less traffic in general (IPv4 and UDP) also receive proportionally less TCP traffic. The same addresses receiving less TCP probes as well as less UDP and IPv4 packets strengthens the theory that the address ranges were temporary unavailable or allocated during the collection period.

The similarity between the IPv4 and TCP data allow for the same conclusion. Namely, the subnets of the TU Delft are targeted equally in terms of TCP probes meaning that the majority of the addresses received the same amount of traffic and that the attackers do not prefer some address ranges over others.

TYPE OF PACKETS

TCP is a connection oriented protocol and therefore has the need to initiate and close connections. The management of a connection is done by setting the corresponding header flag values SYN and FIN. Therefore, it is interesting to see which in which amount the different flags are sent in the 631,424,425 collected probes.

Table 5.3 shows that three type of flag combinations are set in 98.54% of the received traffic. The SYN packets consist of 87.97% of the received TCP traffic. This is an expected percentage as these are the initialization packets which are sent for the establishment of a TCP connection. However a considerable amount of traffic having the SYN and ACK flag set are received. This indicates possible backscatter, which has also been observed in packets using the UDP.

In addition, the percentage of packets received having the RST flag is low (0.78%). However in absolute numbers this percentage can be seen as quite significant, nearly 5 million packets. The explanation lies in the fact that these types of packets are also used in a particular scanning technique called stealth scanning.

The analysis of the flags set in the captured traffic has shown that the majority has only the SYN flag set. Also the flags indicate possible backscatter. Therefore in the next sections of this thesis only the TCP traffic having the SYN flag set will be analysed as potential network scans employ TCP-SYN packets.

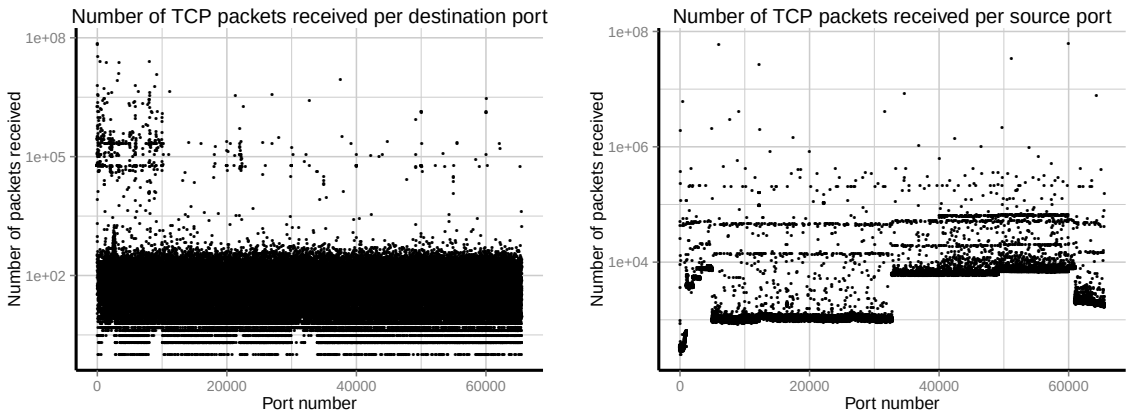
TARGETED DESTINATION ADDRESSES (TCP SYN)

By filtering traffic other than TCP-SYN the destination address analysis should be repeated in order to verify that the destination addresses are targeted uniformly. Figure 5.1 shows the cumulative distribution of number of addresses receiving a TCP packets having the SYN flag set in purple. The variation in number of packets received per address is significantly reduced, manifested by the steepness of the curve, compared to the curve representing the total number of TCP packets sent in green.

Figure A.1d in Appendix A showing the number of packets received per destination address also confirms the reduction in variance. Although figure A.1d still reveals that some address ranges are probed less frequently than others, the majority (roughly 95%) receive the same amount of traffic.

Filtering out other traffic than TCP packets having only the SYN flag set results in a near uniform distribution of packets received per destination address. The lack of full uniformity is partially caused by addresses receiving less packets (roughly 5%) due to, as for the other traffic (UDP, IPv4), the allocation or the temporary unavailability of these destination addresses. Manual inspection of the data shows that the seven addresses receiving an abundant amount of traffic have suffered an attack or misconfiguration as the packets are sent from the same source address towards the same destination port. Therefore, one can conclude that the attackers do not prefer some ranges of the subnet over others.

DESTINATION PORTS



(a) TCP SYN packets received by destination port

(b) TCP SYN packets received by source port

Figure 5.3: Number of TCP SYN packets received per Port

Destination port number	Service	Number of packets
22	Secure Shell	71,457,750 (12.78%)
23	Telnet	66,516,565 (11.90%)
80	Hyper Text Transfer Protocol	33,954,563 (6.07%)
8,080	Hyper Text Transfer Protocol (Alternative) *	25,795,195 (4.61%)
443	HTTP over TLS SSL	25,520,471 (4.56%)
3,389	Windows Based Terminal	24,250,717 (4.33%)
1,433	Microsoft SQL	24,011,892 (4.29%)
3,306	MySQL	12,210,529 (2.18%)
9,200	WAP connectionless session service	12,084,324 (2.16%)
37,564	**Unassigned	8,931,429 (1.59%)
Total	-	304,733,435 (54.53%)

Table 5.5: Top 10 destination ports, in terms of received TCP SYN packets (percentage)

* Mainly Tomcat server
**Likely port used for setting up a proxy

Figure 5.3a shows the number of TCP SYN probes received per destination port. The majority of destination ports have received a similar amount of packets, on average a port received in the order of 100 packets during the collection period.

However as with the UDP traffic, some ports receive more traffic than others, indicating a more active search for specific ports. Table 5.5 shows that only ten ports are responsible for more than half of the received traffic. Confirming that scanners in general have a preference in the targeted ports.

Source port number	Number of packets
60,000	61,649,414 (11.03%)
6,000	59,864,043 (10.71%)
51,158	34,485,572 (6.17%)
12,200	26,796,196 (4.79%)
34,680	8,263,152 (1.47%)
Total	191,058,377 (34.19%)

Table 5.4: Top 5 source ports, in terms of sending TCP SYN packets (percentage)

Inspecting the top ten of scanned ports brings insights into the special interest of scanners for these ports and their associated services. The most scanned ports 22 (ssh server) and 23 (telnet), both are coupled with services allowing to control the system. As with the most scanned UDP ports, services using TCP ports in Table 5.5 have suffered from various security bugs and are prone to vulnerabilities.

The variation in packets received per destination port is to be expected as the scanners focus on ports associated with services as specified by the IANA. In addition scanners also target ports not specified by the IANA but which are often the default ports of services such as MySQL and WAP. This is reflected by the data, where 97% of the ports are probed less than 300 times, and less than 1% receive between 60,000 and 71 million probes. The difference in number of packets received shows that scanners target specific ports.

SOURCE PORTS

In the analysis of UDP packets, the source ports from which the packets have been sent are analysed. This shows that some source ports are used more frequently to send packets. This is also the case for the captured TCP data. Figure 5.3 shows the same kind of division between packets sent from different ranges of source ports. In the lower range of source port (ranging from 1 till 30,000), on average 1,000 packets are sent whereas in the higher range (from 30,000 till 65,000) on average 10,000 packets are sent per port.

In addition, as with the collected UDP traffic, some source ports are preferred over others for sending traffic. Table 5.4 shows that five source ports are responsible for sending 34.19% of the data. This confirms that scanners prefer the use of some source ports over others. The usage of particular source ports over others can be caused by the usage of different scanning tools.

OTHER TCP HEADER VALUES

A deeper analysis of the different values in the headers of the TCP packages reveals other anomalies. TCP SYN packets should have specific or random values filled in for some header fields. Therefore these fields give an insight if the captured traffic has the expected values or that some packets are crafted preferring some values over others.

Window size As specified by the RFC describing the TCP [35], the window size of the TCP SYN packets should be set to 0. This is confirmed by the analysis of the window sizes, which shows that all captured packets have the window size set to 0.

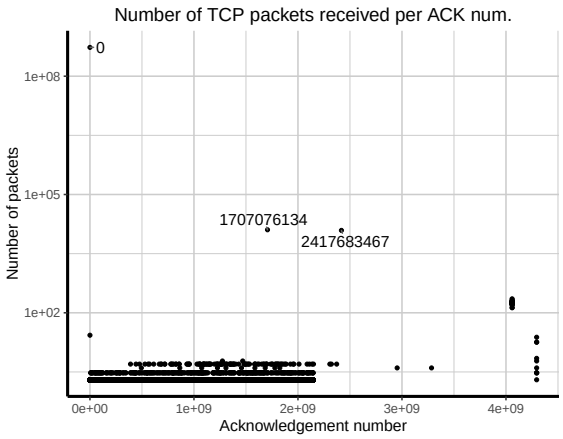


Figure 5.4: Distribution of received acknowledgement numbers

Acknowledgement number The acknowledgement number should also be 0 as specified by the RFC describing TCP [35]. This holds for the majority of the data, where 96.25% of the received TCP SYN packets have an acknowledgement number of 0. Figure 5.4 shows that there are two acknowledgement numbers, 1,707,076,134 and 2,417,683,467, which in addition to 0 are frequently found in the data. Also figure 5.4 indicates that the range of 4,059,102,497 till 4,059,102,528 is a preferred acknowledgement number value. In the range, 33 acknowledgements numbers are found in over 100 packets, while other acknowledgement values, other than 0, are not present in more than 5 packets.

The abundance of specific acknowledgement number can indicated the existence of a scanning tool. The tool could use the acknowledgement number as a filter for incoming packets to distinguish regular traffic from response to sent probes.

Sequence number The sequence number of a TCP SYN packets should be random as is specified in [35]. Assuming that the selection of a sequence number happens according to a uniform distribution, and knowing that the there are 2^{32} possible sequence numbers and that only 2^{29} packets were recorded, it is highly unlikely that a large number of packets have the same sequence number.

Sequence number	Number of packets
30,000	4,620,620 (0.82%)
2,406,000,322	2,506,123 (0.44%)
1,112,425,812	2,303,564 (0.41%)
0	1,936,146 (0.34%)
113,208,957	946,410 (0.16%)
Total	(2.17%)

Table 5.6: Top 5 sequence numbers, in terms of sending TCP SYN packets (percentage)

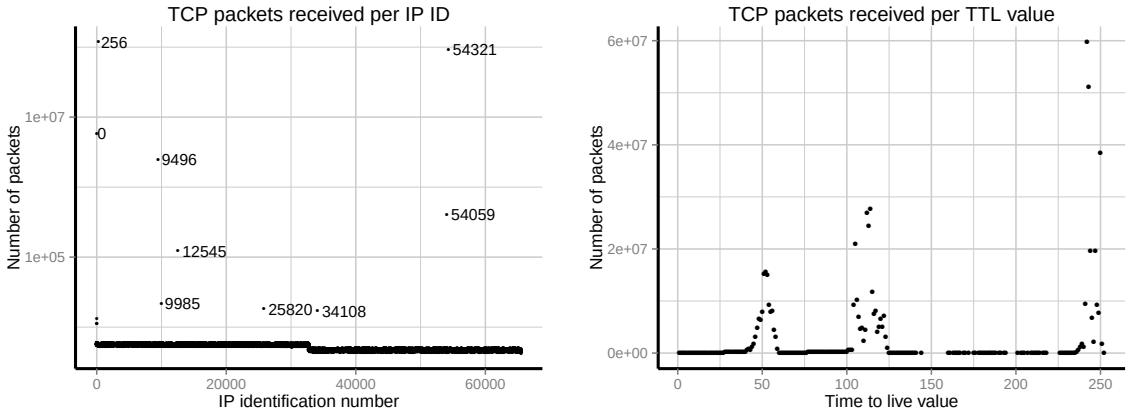
The probability of a packet having a sequence number S_n is $\frac{1}{2^{32}}$ as there are 2^{32} possible sequence numbers and they are randomly generated. The probability of receiving n packets with the same sequence number S_n is equal to $\frac{1}{(2^{32})^n} * (2^{29})^n$ when $\frac{2^{32}-1}{2^{32}}$ is taken to be equal to 1. Using the formula to compute the probability of two packets having the same sequence number results in a probability of less than 0.02, for three packets the probability is reduced to 0.002. From a manual inspection of the dataset, some sequence numbers reoccur more than twice, suggesting that some sequence numbers are preferred over others.

Table 5.6 confirms this suspicion, showing that five sequence numbers can be found in 2.17% of the recorded traffic.

The windows sizes in the captured TCP SYN packets shows no sign of package crafting. However the Acknowledgement numbers and the Sequence numbers do, as some numbers are abundantly present in the captures probes. As will later be explained in the thesis the sequence number and acknowledgement numbers are used by scanning tool to distinguish incoming traffic from responses of sent scanning probes.

IPv4 HEADER VALUES

The TCP header values revealed that the collected packets do not have the expected random header values. Although the IPv4 header does not contain the same amount of information and we only look at TCP traffic, it would be interesting if some values are preferred for IPv4 header fields. In the IPv4 header there are two interesting fields, the IP identification number, which should be random and the time to live, which depends on the system implementation.



(a) Distribution of received IP identification numbers

(b) Distribution of received IP identification numbers

Figure 5.5: Distribution of IPv4 header values

IP identification number As mentioned, the IP identification number is random for every TCP SYN packet. Therefore the collected numbers from the headers of the received packets should spread uniformly among values between 0 and 65,535.

Figure 5.5a shows the number of packets which were received with a certain IP identification number. As assumed, the distribution follows a uniform distribution. A slight difference of received number of packets can be noted between identification numbers below and above 32,500 where the number of packets received is measured to be roughly 5,500 and 4,500 respectively. Besides the variation in number of packets, as with the sequence numbers and other header values it seems that some values are preferred over others. The labels besides the points in figure 5.5a show the IP identification number. This demonstrates that there are 11 IP identification numbers which have a higher occurrence in the dataset, suggesting that these numbers are preferred by some scanners.

Time to live The time to live (TTL) usually depends on the operating system from which the packets originates. This was shown by Beverly [45] which conducted research on determining the OS a packet originated from. In his research he states that the OS can be determined using only header values including the time to live from IP/TCP packets. This is confirmed by documentation of "p0f" [46] which is a widely used OS fingerprinting tool.

Figure 5.5b shows that there are three peaks noticeable. These peaks coincide with the known values for the different operating systems where the values 64, 128 and 255 are generally used by respectively Mac, Windows, and Linux systems. Therefore from a TTL value perspective the data seems normal. In addition, no records have been found having TTL values equal to zero meaning that there are no packets in the dataset which originates from true unroutable data.

The Time to live values offer an insight into the used operating systems from which scanning probes are sent. The majority of scanners use the Linux operating system. In addition the distribution of IP identification numbers in the collected packets also reveal that used scanning tools modify the IPv4 header to alter the IP identification numbers. As later in the thesis will be explained the alteration of IP identification numbers is a trait of certain scanning tools.

6

IDENTIFICATION OF SCANNING TOOLS

This chapter will analyse different tools available for network scanning. The discussed tools may provide additional information which can be used to fingerprint scanners. Research performed by Durumeric et al. [29] suggests that some tools are used more to scan a certain port number, supporting the hypothesis that different tools are used to discover different services. Even though the thesis focusses on scanners using TCP SYN packets, for completeness most commonly used scanning techniques are presented in the following section. Following the different scanning techniques, a closer look at the different tools used to perform TCP SYN scans can be found. Subsequently an analysis of the usage of the presented tools is performed.

KNOWN SCANNING TECHNIQUES

ICMP

The first scanning technique presented uses ICMP message. This method uses ICMP packets, also known as *ping* messages, and only require the IP protocol. An echo request packet is sent from the scanner to the scanned devices which in turn responds with an echo reply. This style of probing bring little information about the scanned device, often only revealing if the device is connected to the network or not. This is due to using the IPv4 protocol which carries less information than other higher layer protocols such as TCP or UDP. Furthermore firewall rules may prevent the scanner from receiving an echo reply resulting in the scanner not knowing if the device is offline or behind a firewall.

ICMP messages can be crafted in order to gather more information about the scanned devices. Arkin [47] describes several methods to acquire additional information on the targeted machine using ICMP messages. Sending different ICMP queries to a target can reveal information about the device type (server or router), and which operating system is used.

However this technique remains, as shown in previous section by the low amount of captured ICMP messages and by related work, unpopular.

UDP

Using the UDP protocol for scanning has some advantages over ICMP scanning. First it allows to uncover possibly opened ports on devices which are listening to the UDP protocol. Also, the scanner can determine, with the use of selected destination ports whether services are running on the scanned devices. As explained by De Vivo et al. [48] UDP port scanning works counter-intuitively. The scanner can only see if UDP ports are closed, thereby informing him if ports are possibly open. To elaborate, a scanner sends an empty UDP packet to the port of interest. The scanned device may respond in two manners. If the port is open, the device will not respond as UDP is connectionless and no request has been made by the scanner as the packet is empty. If the port is closed the scanned device will send an ICMP message stating that the port is closed. These responses allow the scanner to deduce which ports are closed, therefore which are probably open. However there are some limitations. The administrator of the scanned device may have disabled sending of ICMP messages indicating closed ports, misleading the scanner about the actual state of the scanned ports.

This leads to a second manner of UDP scanning where scanners, crafting service appropriate packets to trigger a genuine response from the scanned device. This is however a more resource intensive manner of scanning as it generates a high amount of traffic on the scanner side.

TCP

TCP scanning is, as shown in the related work [37] and by the collected data, preferred over ICMP and UDP scanning. It offers more possibilities than UDP because there are more options available in the header which can be used for different types of scans. Additional the TCP behaviour is standardized meaning that for every type of packets sent to a port, whether it is closed or open the response is known. In general the TCP scanning techniques are sorted into three categories, connection, stealth, and SYN scans.

Connection The connection type scan is the most elementary probing technique of the available TCP scanning methods. It tries to establish a TCP connection on the targeted port with the device it is scanning. This is done through the regular connection establishment method described in the TCP protocol [3]. When the connection is established, then the port is open, else the port is closed.

SYN SYN scans are the most prevalent in the used dataset, and are therefore the studied data in the thesis. These types of scan use the connection property of TCP. The technique is similar to the connection technique. However, instead of creating a connection on the targeted devices for specific ports, the scanner crafts TCP SYN packets and sends them to the targeted devices' ports. By crafting the packet and not using the standard TCP stack from the OS, a scanner does not have to wait for an acknowledgement before reopening another connection and may therefore send a series SYN packets one after another. While sending the packets the scanner waits for incoming acknowledgements. If the target acknowledges the request sent by the scanner, the scanner knows the port is open. Knowing that the port is open is enough for the scanner and therefore eliminates the need for the scanner to complete the connection. This eliminates half of the messages needed to be sent by the scanner, thereby increasing the scanning speed. As will be explained in the following section, this method can bring other advantages at the cost of the targeted devices being able to identify the used scanning tool.

Stealth Stealth scans are a group of different types of TCP scanning techniques supposed to stay undetected for the scanned devices and their firewalls. The methodology is similar to that used by UDP scans. The main goal is to trigger a scanned device into sending a TCP reset packet. These scan types are using a property of the TCP which causes closed TCP ports to send RST packets when specific header flags are set. This allows the scanner to determine which ports are closed, thus also which ones are open.

KNOWN SCANNERS

Similar research performed in 2015 by Myers et al. [49] reveals four scanning tools, NMap [1], ZMap [2], Masscan [3], and Unicornscan [4] that can be used to probe the Internet. Various publications by Durumeric et al [29] and Adrian et al. [50] mention these tools which are capable of sending SYN packets, the main focus of the thesis.

In addition to the existence of these programs, the research performed by Durumeric et al. [29] shows that packets received from ZMap and Masscan tools can be identified. This allows to study the behaviour and properties of scanners by the used tools. Therefore an in depth discussion and the detection methods for identifying packets sent by these tools is presented.

ZMAP

ZMap is presented in 2013 by Durumeric et al. [10] and is often employed in research performed on port scanning. Furthermore, Adrian et al. [50] claims that it is the fastest scanner to date. The speed is due to the workings of the scanner, ZMap is stateless and uses different threads to send and receive packets. As mentioned earlier, the SYN type scan does not need to establish a connection, therefore ZMap can send SYN packets in one thread and handle the responses in another one. However, this separation of threads imposes a challenge. ZMap must be able to distinguish responses to the TCP SYN packets from regular traffic and background radiation. In order to achieve this, ZMap uses the TCP sequence number field. When sending a packet to a targeted device, ZMap uses Advanced Encryption Standard (AES) described by Daemen and Rijmen [51] to encrypt the IPv4 source and destination address. The sequence number is then set to this calculated number. Then when a packet is received ZMap, performs the encryption and validates if the TCP

[1] https://nmap.org/
[2] https://zmap.io/
[3] https://github.com/robertdavidgraham/masscan
[4] *At the time of writing the home page of Unicorn scan was unavailable http://unicornscan.org

acknowledgement number is equal to the results of the encryption plus one. If this is the case, the packet is a response to a scan, if not it is discarded as regular traffic.

For the thesis it is interesting to see whether or not packets sent from this scanner can be identified. One method to verify if a scanner is using ZMap would be to break the used encryption on all received packets. However this would not be feasible considering the amount of data which needs to be analysed.

Fortunately, the article presenting ZMap states that the program has a fingerprint. Namely when crafting packets the IPv4 identification field is set to a fixed value of 54,321, which can be verified in the source code of ZMap [5] This means that captured packets having the IP identification value set to 54,321 are likely to be ZMap probes. Specifically, the chance of the packets not being generated by ZMap is $\frac{1}{2^{16}}$.

Therefore, if a scanner sends a lot of packets which are identified as being sent by ZMap, the attacker is likely to use ZMap.

MASSCAN

Masscan is another fast scanner, alike ZMap. Masscan uses a similar technique compared to ZMap, it also generates several threads to send and receive packets, introducing the same challenge of recognizing responses to the sent scan packets. In order to recognize incoming packets Masscan uses SipHash, a hashing algorithm developed by Aumasson and Bernstein [52] to hash the source and destination ports, and IPv4 addresses. This hash is stored into a table when a scan packet is sent. Upon receipt of a packet the source and destination ports and IPv4 addresses are hashed. If the hash is present in the table it is a response of a scan, if not, it is normal traffic or radiation.

For Masscan no values in the IPv4 and TCP header are hard coded. Fortunately Durumeric et al. [29] described a method to detect packets sent by Masscan. Masscan uses an XOR operation on the TCP sequence number and destination port, and IPv4 destination IP. This number if then stored in the IPv4 identification number as shown by the following formula:

$ip\_id = dst\_ip \oplus dst\_port \oplus tcp\_seq$

As the XOR operation can be reversed and all the values are known. When receiving a packet it is possible to check if the packets is crafted by Masscan. Analogous to the detection of scanners using ZMap the randomness of the IPv4 identification field allows for the detection of scanners using Masscan.

UNICORNSCAN

Unicorn scan is a less popular tool, however, it has been analysed by Myers et al. [49]. As ZMap and Masscan the sending and receiving of packets is separated. Yet only one thread is used for receiving and one thread is used for sending the packets, limiting the scanning speed.

Similar to previous mentioned scanning software, Unicornscan faces the same challenge to filter out incoming packets. A presentation dating from 2005 by Lee[53] given at a Defcon 13 [6] reveals how the scanner solves this problem. Similar to ZMap the TCP sequence number is used to store information in the packets sent by Unicornscan. At each start Unicornscan generates a session number, a random 32 bit integer (called $sessionKey$), which is used to generate the TCP sequence number (tcp_seq). To generate the sequence number the following XOR operation is used:

$tcp\_seq = sessionKey \oplus src\_ip \oplus ((src\_port << 16) + dst\_port)$

Here src_ip is the scanner's IP address, src_port is the scanner's source port, and dst_port is the source port on which the receiving (destination host) party is being scanned. When a packet is received by Unicornscan the reversed operation is performed, if the result is the session key then the packet was sent by Unicornscan and is a response to a packet, else it is normal traffic or background noise.

Being able to classify this scanner seems difficult as the author of the program uses a random session key to generate the TCP sequence number. However when analysing the code [7], it becomes apparent that detecting Unicorn scans is possible. The generation of the random sequence key only happens once when a scan is performed, at the start of the program. This implies that during the entire scan the same session key is used. Therefore when the reversed XOR operation is performed on packets sent form a scan using Unicorns

[5] https://github.com/zmap/zmap/blob/de190e7173ef97cd172d3c895f9f2cb181a762e2/src/probe_modules/packet.c
[6] https://www.defcon.org/html/defcon-13/dc13-speakers.html#Lee
[7] http://sourceforge.net/projects/osace/?source=typ_redirect

scan the same session number appears. This would translate into packets, being sent from the same scanner using Unicornscan, all having the same session number.

One could argue that the reversed XOR operation used to find the session number could always be equal to the sequence number, thereby falsely identifying the scanner as using Unicornscan. However the TCP sequence number are randomly generated when a TCP SYN packet is sent. This number is 32 bit long, thus ranging from 0 to 2^{32}, simple probability calculation show that the probability p for a scanning tool, having sent n packets, being misidentified as using Unicornscan is:
$p = (\frac{1}{2^{32}})^n$
Therefore, after receiving only a small amount of packets, the probability to identify Unicornscan increases dramatically.

NMAP

The last analysed scanning tool is NMap [9]. As previously discussed scanner tools, this software allows for TCP SYN scans as described by Lyon [54]. The same problem as with the other scanning tools arises. NMap employs an elaborate encoding scheme. It uses a session cookie, a random port, a number of tries, and a ping sequence which are encoded into the sequence number. If the returned sequence number has theses values encoded the packets is considered to be a response. For more information we refer to the decode function in the code provided on Github [55].

Unfortunately no fields of packets sent by NMAP are hard coded in the header. In addition, research from 2014 [29] states that no known characteristics exist allowing such identification. However, the source code of NMap is available to the public at Github [56]. This allows further investigation, into the packet identification mechanism of NMap. Similar to Unicornscan, NMap also uses a session key called *seqmask* to perform a XOR operation to generate a TCP sequence number. However, in contrast to Unicornscan, NMap uses two more values to perform the XOR operation. The XOR operation is best illustrated by the original function found in the source code of NMap, however a simplified version is depicted in figure 6.1.

The encoding function depicted in figure 6.1 generates the sequence number given to a scan probe. The ping sequence number and try number are variable which record the number of times a destination IP has been pinged and repeated attempts there have been to send a probe. These two variables, are unknown to us whereas we do not know the number of times we have received a packet from a certain IP since we do not know when the scan began.

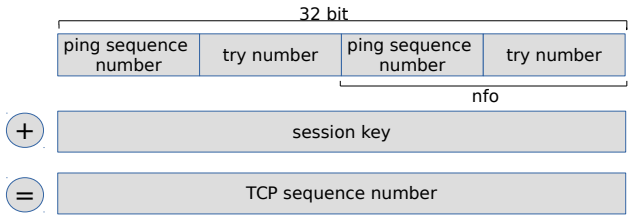


Figure 6.1: NMap TCP sequence number generator

Digging deeper into the code reveals another encoding performed by NMap in the TCP header of probes. If NMap is used with the default settings, no source port is specified, additional information is stored in the source port. The additional information stored in the source port are the ping sequence number and the try number. NMap chooses the source port, when none is specified by the user, also referred to as base port randomly between (33,000) and (65,536 - 256).

Selecting a base port not higher than (65,536 - 256) is done in order to be able to encode, as shown in figure 6.2, a 8bit number consisting of the try number or ping sequence number without causing an overflow. In addition the base port is altered during the scanning procedure, so there lays another challenge. However it is possible to determine the base port and the used session key when one received enough packets.

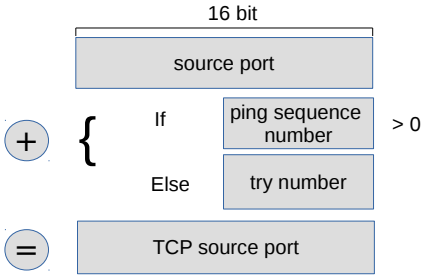


Figure 6.2: NMap TCP source port generation

When sufficient packets are captured a reverse of the encoding algorithm used by NMap to determine

certain variables can be applied. When receiving a packet from a scanner using NMap one has the generated sequence number in which the ping sequence number and try number are encoded with the random session key. So both numbers are obfuscated, however when receiving two packets from the same source IP one can use an interesting property of the XOR operation. When the same key K is used to encode two different values A and B, the XOR operation of these results will give the XOR operation between the two encoded values $A \oplus B$.

Thus when two sequence numbers of a scanner using NMap are received, the XOR operation should be applied. If the same session key is used the outcome is the result of the XOR operation on the encoded ping sequence number and try number of each packet. However the ping sequence number and try number are encoded into the sequence number in a particular manner. As shown in figure 6.1, they are encoded into a variable called nfo, which is then mirrored and thereafter XORed with the session key. By mirroring the nfo value prior to the final XOR operation using the session key, the XOR result of two sequence numbers using the same session will also be mirrored as it is the XOR operation of two mirrored numbers. This is interesting as there are only 65,535 mirrored numbers for every possible result of the XOR operation of two sequence numbers (32 bit in general). This demonstrates that the probability of having a XOR operation of two sequence numbers resulting in a mirrored number is equal to $\frac{1}{2^{16}}$ which is equal to the probability of packets having the same IP identification number.

Hence, in theory if enough packets are sent it is possible to detect NMap provided the program does not alter session keys too often. This assumption has been tested, by scanning a range of the TU using NMap with the default settings. During the scanning NMap changed session keys, however the number of packets related to each session number was high enough to eliminate coincidence.

In addition, besides being able to detect possible scanners using NMap, the code shows that it is also possible to detect the base ports, revealing which try number and ping sequence number are used by the program. This is due to the fact that NMap stores the ping sequence number or try number in the base port as shown in figure 6.2. When combining this finding with the results of XORing the sequence numbers, it is possible to calculate the base port used during a scan performed by NMap. However, such analysis is beyond the scope of this thesis.

CONFIDENCE LEVELS AND INTERVALS

All of the methods to detect scanning tools rely on the randomness of numbers found in the packet headers. Therefore when receiving the same numbers from a scanner for all its packets it is reasonable to assume that a specific tool is being used. However, because most values are randomly chosen, it could be that the numbers originating from a scanner are truly random and look as if they are not. This is the case when receiving a low number of packets, however when receiving a lot of packets the probability of this happening is very low. The problem in this assumption is that we do not know the exact number of packets necessary to confidently state if a scanner is using a certain tool. Accordingly a number of packets should be examined for which with a certain degree of confidence can be confirmed that indeed a scanning tool is being used. In order to calculate such certainty, statistical analysis is employed.

The basic method of calculating the required number of packets would be to assume that the header values in those packets are uniformly distributed. One may calculate the probability of the scanner sending the same random header values in n packets. The random numbers r in the header values, which are used to detect scanners are either 16 or 32 bits. This means that the probability p of a scanner sending n packets having the same random header value is: $p = \frac{1}{r^n}$. The probability of recording ten packets having the same random 16 or 32 bit header is respectively $6.8 * 10^{-49}$ and $4.7 * 10^{-97}$. As these values are very small, one may assume that recording ten or even five packets with the same number is enough for identifying a tool. However this simple method does not offer a clear measure certainty and is arguable, therefore the calculation of confidence interval is necessary.

Confidence intervals are employed to estimate probabilities of datasets. The calculated probability resides within intervals delimiting its minimum and maximal value. Additionally a confidence level is attributed to the probability which indicates the certainty of the probability lying within the interval.

In order to calculate the number of probes required to flag a source address as sending probes generated by a certain tool a distribution needs to be derived. Scanning tools are detectable through the use of specific header values, which under normal circumstance are random. Thus for each packet sent from a source address it can be flagged as generated by a scanning tool. Whether or not a packet sent is generated by a certain

tool translates into a binomial distribution. Calculating a confidence interval for a binomial distribution results in the number of probes which need to be flagged in order to confirm that a source address sends probes using a specific tool.

The most widely used confidence interval calculator for the Binomial distribution is using a normal approximation, also known as the Wald interval. The normal approximation method is widely used as described by Brown et al. in 2001 [57]. Further research reveals that the normal approximation method is not accurate when it comes to calculating intervals for probabilities close to 0 or 1. The ability to perform calculations using probabilities close to 1 is necessary for calculating confidence intervals and levels in this case. The goal is to determine with a probability close to 1 that a certain scanning tool was used to perform a scan. In addition, the normal approximation method has varying calculation times as described by the authors, which is not a desirable property when performing calculations over large sets of data.

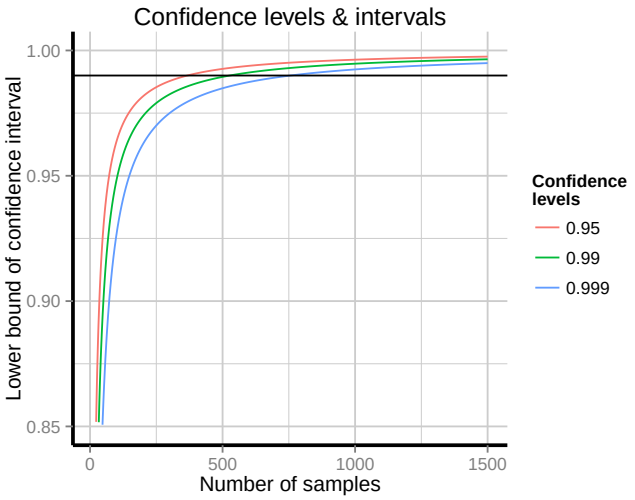


Figure 6.3: Lower confidence bound per confidence level for given number of samples

Another option is the Clopper & Pearson (CP) method as described by Clopper and Pearson [58]. The CP method of calculating the confidence interval is correct, meaning that the confidence values are true for values near zero and one. Using the CP algorithm the number of packets which need to be flagged as originating from a certain tool can be determined for a certain confidence interval.

The confidence interval is not sufficient, the confidence level should also be determined. The confidence level indicates with which certainty can be confirmed that the probability lies within the calculated confidence interval..

The calculations have been performed and the results are depicted in figure 6.3. The figure clearly underlines the initial intuition that relatively little packets are needed to identify the tool used by the scanner. The horizontal line is set at .99 which is a relatively high lower confidence bound. This level is chosen as the minimum accepted confidence bound. Three confidence levels are depicted, .95, .99, and .999. In order to be absolutely confident that the scanner is using a certain tool the confidence level of .999 is chosen. The chosen confidence level leads to 750 packets which are needed to confirm if a scanner is using a tool. For ease of use in the following section, a number of 1000 packets are used.

ZMAP LIKE

ZMap-like scanners are scanners which alike ZMap use a specific IP identification number (ID) number. ZMap uses an IP ID number of 54321, however it is possible that people altered the source code and replaced this fixed number for another one. One possibility is that the individual altering the source code has chosen another fixed number. It is also likely that the individual altering the source code has obfuscated it a bit more and has set the ID to a pseudo random number or even a random number. The first realm can be detected as the number would be static, these events can be classified as ZMap like scanners. It could also be that the used scanner is not an alteration of ZMap but simply a scanning tool which also sets the ID to a fixed number. Even though the scanner does not use the ZMap software, it still belongs to the ZMap like class, as the fingerprint is the same.

Flagging scanners using ZMap like software is more elaborate and requires retrieving extra information besides just looking at packets sent and comparing the packets for the same identification number. As

IP ID	Number of detected scanners	Number of packets	Scanners with distance below 256
0	633 (12.60%)	4,316,833 (2.02%)	62 (9.79%)
1	1 (0.01%)	1,891 (0.00%)	-
256	2,922 (58.19%)	115,567,225 (54.11%)	2543 (87.02%)
9496	298 (5.93%)	704019 (0.33%)	3 (1.00%)
9985	1 (0.01%)	16,279 (0.01%)	-
12545	1 (0.01%)	117,204 (0.05%)	-
54059	1 (0.01%)	401,649 (0.19%)	-
54321	1,164 (23.18%)	92,437,035 (43.28%)	927 (79.63%)
Total	5021 (0.44%)	213,562,135 (38.21%)	-

Table 6.1: ZMap like scanner per ip id

demonstrated in the previous section in order to be able to validate a scanner using ZMap like software, a minimum amount of packets is needed. Previous section argues that the number of packets should be 1000 because it provides 0.999 percent certainty that the probability of the scanner using ZMap like software is greater than 0.99.

When running the detection software on the data there are in total 5,021 scanners having sent more than 1,000 packets which have been detected using ZMap like scanning software. It is to be noted that from the 1,135,380 IP addresses only 0.44% are recognized as using an ZMap like tool. These scanners are responsible for sending 213,562,135 packets which is 38.21% of the total amount of packets received. As shown by table 6.1 some ID numbers are preferred over others.

Highly interesting is that the identification number of 256 is more popular than the standard 54321 from ZMap. However both these scanning tools are responsible for 97.38% of the traffic generated by ZMap like scanners. Also some lone scanners are visible, four identification numbers are only used by one scanner, indicating some kind of private tool.

Also noteworthy is to see whether there are differences between the scanners using the same tool. Analysing the distances (number of addresses between addresses) between the used IP addresses discloses a notable difference between scanners using different identification numbers. A striking phenomenon being the rather large distance between IP addresses for ID 0 and 9,496 However for the numbers 256 and 54,321 there are a lot of addresses which are adjacent, therefore having a low distance. The difference in distances between source addresses suggests that there is a difference between the usage of the tools with different identification numbers. One could argue that the small distance between addresses using the same tool is a sign that the tool is popular in some parts of the world as IP addresses within a distance of 256 are often in the same geographical region. Another hypothesis is that scanners use multiple adjacent addresses to perform scanning reducing the measured distance of addresses using these tools.

MASSCAN LIKE

As described previously Masscan uses a XOR operation to store the destination, port and IP and the sequence number, allowing the detection of the scanning tool. A scanner is flagged as using Masscan if at least 1000 packets are received and identified as generated by the tool.

In total there are 2,520 IP addresses having sent packets which are identified as crafted by Masscan like software. However from those 2,520 not all the packets sent, identify with the algorithm used by Masscan. Only 264 of the addresses have 100% of their packets originating from Masscan. In addition the number of packets sent by these scanners needs to be looked at, as demonstrated earlier a minimum amount of 1000 packets is needed in order to confidently determine that a scanner is using a tool. From those 264 scanners only 150 have sent more than 1000 packets, which greatly reduces the number of scanners using Masscan. Implying that from all the recorded source addresses, only 0.01% are using Masscan. However when looking at the generated traffic we find that these scanners are responsible for 65,464,216 packets which is 11.74% of the traffic.

With the scanners using Masscan, the distance between the IP addresses is also large, only 15.33% of the scanners has a distance less than 256. Since this is a small number, it is easily noticeable that two companies form the 15.33% of close IP addresses. Namely Errata Security which uses 209.126.230.71/75 block and Net System Research using the 169.54.233.118/121 block to perform scans for research purposes.

UNICORNSCAN LIKE

For unicorn scan the same method is used as with previous scanners. A minimum amount of 1,000 is needed in order to classify a scanner as using Unicornscan. Also all the packets sent must use the same cookie.

There are in total 259 scanners identified as using Unicornscan, which is 0.02% of all addresses. These scanners are responsible for sending 4,980,426 packets which is only 0.88% of the total amount of packets sent.

The number of addresses with a small distance is low, only 3 addresses of the 259 have a distance smaller than 256 which amounts to 1.15%. As shown by scanners using Masscan, the closeness of IP addresses can be an indication of the addresses being used by the same scanner. As the numbers suggest there are no adjacent addresses using Unicornscan. Nevertheless Unicornscan offer additional information, it offers a cookie which is generated randomly. The different cookie form the scanners can be compared in order to see whether some cookies occur more often then others. There are no two addresses using the same cookie.

NMAP LIKE

As explained in the previous section, the procedure used to detect scanners using NMap is more complex than the one used for detecting previously discussed scanners. This is due to the possibility that the session key used by NMap may alter during the scanning period. To that end in order to stay within the certainty values defined in the beginning of this section a scanner is identified as NMap like if the number of packets sent corresponding to one session key is equal or larger than 1,000. Subsequently, scanners using a fixed sequence number will also fall under the NMap like classification.

This selection criteria results into 1,366 addresses, which is 0.12% of the total address having sent packets to the TU. These scanners are responsible for 13,143,213 packets which is 2.35% of the total traffic. The distance between the addresses is also calculated, only 88 (6.44%) of the addresses have a distance below 256. Beside the distance, the number of time the scanners have changed their session key is also scarce. From the 1,366 scanners, 9 have changed their session key, whereas the others have not.

7

SPACIAL ANALYSIS

This section presents the spacial analysis of IP addresses sending probes to the network of the TU Delft. Spacial analysis might yield information about the scanners, such as preferences of IP ranges or countries. The topics are, in order: the scanners according to the entire IPv4 space, the scanners location presented country wise, scanner activity related to AS numbers.

ORIGIN OF SCANS

In order to get a better understanding of which IP addresses scan the TU Delft, it is important to see how they are spread in the IPv4 space. Clustering of IP addresses could indicate that scanners prefer some IPv4 ranges over others to perform scans.

One way of approaching such analysis would be to plot each IPv4 address having sent a packet to the TU Delft network. However this is not feasible, as the IPv4 address range is 2^{32} which would require an image of at least 65 536 by 65 536 pixels to represent each IPv4 address. Therefore it is necessary to reduce the IPv4 space to /24 subnets, each containing 256 IP addresses. This reduces the image size to 4096 by 4096 pixels [1].

Furthermore, in order to respect the spacial properties of IPv4 addresses, a special plot called the Hilbert curve is used. The curve plots IP addresses belonging to the same subnet near each other. This Hilbert curve is inspired by the comic XKCD written by Munroe [59] and used by an anonymous researcher [60] in 2012 to present a census of the internet using the Carna Botnet.

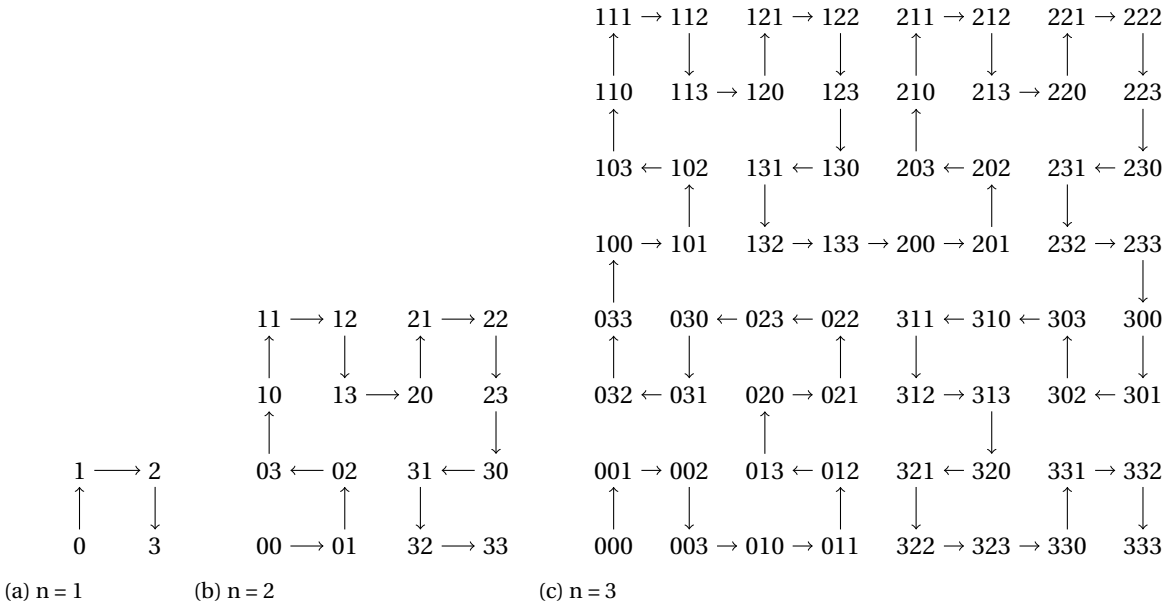
Figure 7.1 shows how the Hilbert curve plots. The alignment allows keeping the spacial properties of the IPv4 addresses. In particular the figure shows the mapping of addresses similar to IPv4 addresses.

In 7.1a the figure shows the mapping for an address range of 4, which can be seen as one times two bits. Figure 7.1b shows the mapping for 16 addresses 16, which can be seen as two times two bits. The first integer represents the first two bits and the second integer represent the second two bits, similar to the dot notation of IPv4 addresses. It becomes apparent that the special positioning of consecutive points by the Hilbert curve preserves the spacial aspect of the addresses. Figure 7.1c shows the mapping for 64 addresses, which can be seen as three times two bits. As with the 16 addresses, the spacial aspect of the addresses is preserved. This also holds for other larger address ranges, thus also for the IPv4 range.

Figure 7.2 shows an IPv4 heat map constructed using the Hilbert curve. The figure shows the number of IPv4 addresses having sent a TCP SYN packet to the university. In addition, an overlay providing various labels for various prefixes owned by known instances is provided. It becomes clear from the figure that the addresses used for scanning are not uniformly distributed. Some ranges are preferred over others. More so the heat map indicates that some /16 clusters have a very high activity when it comes to IPv4 having sent packets to the TU Delft.

This finding could indicate that scanners are using several IPv4 addresses or that some blocks are more prone to being used by scanners. A theory advances that some countries are more likely to host scanners due

[1] Remark from the author: Please print this thesis in highest possible resolution. Best viewed online.



(a) n = 1 (b) n = 2 (c) n = 3

Figure 7.1: Illustration of the Hilber Curver for address spaces of 2,3, and 4 bits.

to legislation or other preponderances. Therefore it is relevant to see whether some countries are more prone to perform scans than others.

COUNTRIES

As mentioned in the previous section, IPv4 addresses are not uniformly spread over the entire address space. This indicates that some countries are more prone to host scanners, as the distribution of IPv4 addresses is also moderately tied to countries. This link of IPv4 addresses and countries is due to assigning of IPv4 blocks to Local Internet Registries (LIR) which are often geographically bound to a country or region.

NUMBER OF ADDRESSES

In order to get the geographical representation of each address having sent a packet to the TU, it is necessary to get the associated longitude and latitude with every IPv4 address. This is done using free data provided by Maxmind [61], specially the GeoLite City database.

From the database one can uncover the locations of 1,111,399 of the addresses having scanned the TU Delft range. This means that from the original 1,135,380 addresses having scanned the collection range, only 23,981 (2.11%) could not be located geographically. The missing addresses could not be found in the database, therefore no coordinate were available for them. The addresses for which coordinates were found in the database are represented on the world map shown in figure 7.3. It becomes visible from the map that the majority of the

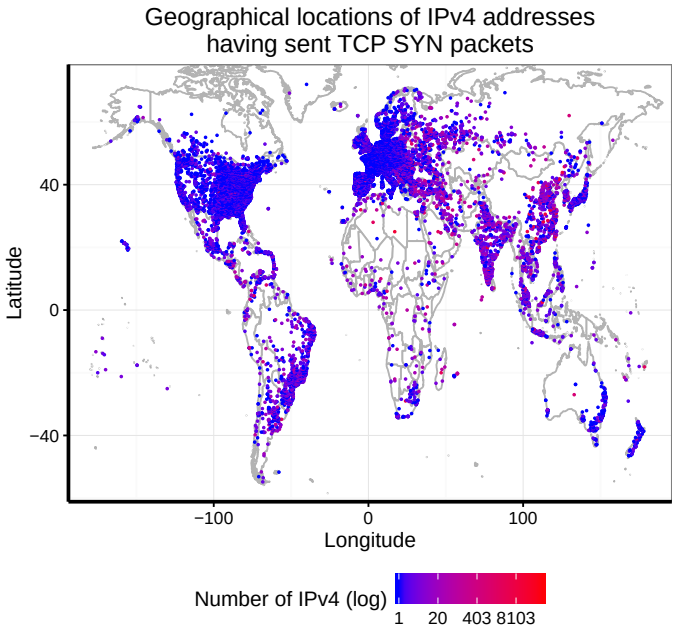


Figure 7.3: Geographical location of IPv4 addresses having sent TCP SYN packets

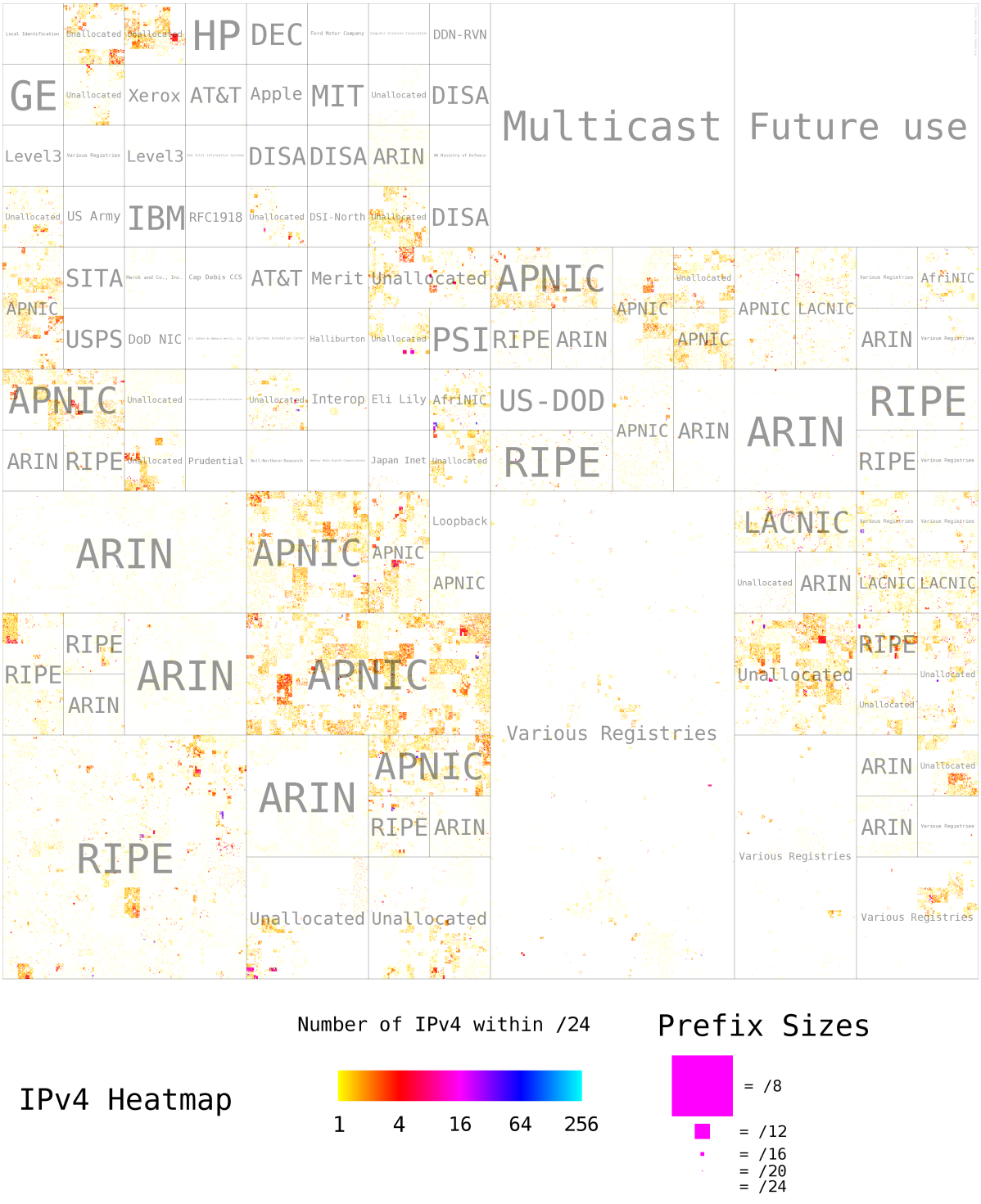


Figure 7.2: IPv4 heatmap of IPv4 having sent packets

Place	Number of IPv4 addresses	*Percentage of IPv4	Number of packets
1	China 429,663 (37.84%)	Libya 4.48%	China 252,762,930 (45.23%)
2	Russia 62,307 (5.48%)	Moldova 2.56%	United States 73,510,239 (13.15%)
3	Taiwan 57,120 (5.03%)	Madagascar 1.87%	The Netherlands 49,014,062 (8.77%)
4	India 56,380 (4.96%)	Fiji 1.40%	Taiwan 29,476,097 (5.27%)
5	Brazil 52,996 (4.66%)	Christmas Island 1.17%	Hong Kong 19,169,605 (3.43%)
6	Thailand 43,471 (3.8%)	Tajikistan 1.13%	Germany 14,401,129 (2.57%)
7	United States 33,933 (2.98%)	Yemen 0.75%	Russia 12,741,955(2.28%)
8	Turkey 33,855 (2.98%)	Albania 0.55%	Republic of Korea 1,159,2211 (2.07%)
9	Moldova 30,988 (2.72%)	Thailand 0.48%	France 8,260,527 (1.47%)
10	Hong Kong 21,285 (1.87%)	Macau SAR China 0.41%	Poland 8,148,739 (1.45%)
Total	821,998 (72.39%)	-	479,077,494 (85.73%)

Place	Average packets per IPv4 address	**Ratio IPv4 address
1	Iceland 25,553	Moldova 8.71 10^{-3}
2	The Netherlands 21,668	Singapore 7.20 10^{-3}
3	Latvia 9,667	Uruguay 4.89 10^{-3}
4	Switzerland 5,041	Hong Kong SAR China 2.93 10^{-3}
5	Seychelles 4,344	Curacao 2.82 10^{-3}
6	Panama 3,355	Belarus 2.54 10^{-3}
7	Lao PDR 2,909	New Zeland 2.48 10^{-3}
8	Germany 2,683	MACAO SAR China 2.46 10^{-3}
9	United States 2,166	Libya 2 10^{-3}
10	Zimbabwe 2,090	Fiji 2.32 10^{-3}
Total	523	-

Table 7.1: Top 10 countries showing scanning behaviour
* Percentage of IPv4 addresses of a country
**Number of addresses used divided by the population

addresses having sent TCP SYN packets are located in developed regions.

Even though the spread of IPv4 addresses having sent TCP SYN packets to the TU Delft on the world map indicates that various regions scan more frequently than others, further analysis is required. Some factors may influence the partition of the scanners, the most obvious being the availability of an Internet connection. Therefore these biases must be taken into consideration. During the analysis of the total number of IPv4 addresses per region, defined in the database by longitudinal and lateral coordinates, a number of inconsistencies are observed. Some of the addresses, most often in rural areas, are given the coordinates of the closest hub, or addresses which could not be geographically located are mapped on the capital of the country or nearest large city. These inconsistencies lead to further investigating regarding the accuracy and reliability of the database. Study performed by Poese et al. [62] reveals that the usage of address location databases is inaccurate when it comes to exact coordinates but is accurate when it comes to countries. Therefore further geographical analysis is done on a country level.

Plotting the number of IPv4 source addresses involved in scanning the network on a country scale, shows as figure 7.3 does, that the number of IPv4 addresses are not uniformly spread over the countries. This leads to the assumption, that some countries are more focussed on scanning than others, or that scanners prefer certain countries over others. Table 7.1 shows that 37.84% of the IPv4 addresses having scanned the TU originate from China. Furthermore, the top 10 scanning countries of the 212 mapped are responsible for 72.39% of the IPv4 addresses having scanned the range of the TU. This supports the notion that the majority of scanning probes originate from a few countries.

The previously used metrics to determine the most active scanning country are absolute such as number of packets and addresses. That is that the numbers are not corrected for any form of possible bias such as number of available connections or population. China has a large number of IPv4 addresses allocated, so do the other countries presented in table 7.1. Also the number of inhabitants of these countries might state an important role in the number of IPv4 address having scanned the range of the TU Delft. Therefore, correction

is needed to get a more representative view of the relative number of scanners per country. In order to achieve this compensation, the number of IPv4 addresses having scanned the university's range is corrected using the total number of IPv4 addresses a country has. This correction will give an indication of the percentage of IPv4 addresses of a country used to perform scans.

Mapping the percentage of IPv4 addresses of the countries involved in scanning the network reveals a different distribution of scanning activities on a country scale. Revealing the contrast between the absolute number of addresses scanning from a country and the percentage of addresses originating from that country.

Table 7.1 confirms the controversy between the two world maps. The table presents the top ten countries with the highest percentage of IPv4 addresses used to perform scans. This indicates that the countries having the most IPv4 addresses scanning are perse the countries having a relatively high percentage of scanning IPv4 addresses. However some countries are present in both columns such as Moldova and Thailand. This information could indicate the relation between the number of IPv4 a country has and the percentage used to perform scans. One could argue that the laws of a countries such as Moldova and Thailand regarding scanning or that providers operating in the country do not sanction port scanning. Looking at the data using different plotting techniques and metric suggests that there is no apparent link.

As the data shows, the number of IPv4 give an insight into the number of scanners, but it does not provide a view of the intensity of the scanners from the countries. In this study case, being the number of packets used to scan the TU Delft. From the data we can observe that not all scanners are equal, the fact being that some send more packets than others. Certain countries might send more packets using less IPv4 than others. Therefore it is necessary to also look at the number of packets sent per country. This sheds light on the scale of scanning performed per country and not only the harbouring of scanners per country.

The different normalizations applied on the data in this section show a common, yet disturbing façade of statistics. Namely the possibility to present data in such a manner that the outcome leads to one conclusion in favour of another one If no normalization is applied, one would be inclined to point at China being the biggest threat when it comes to scanning as it is the country from where most addresses and traffic originate. However, when normalization is applied, revealing the percentage of addresses scanning originating from a country, China is not in the top 10. Normalizing the number of addresses sending probes to the TU by the population of countries also show that China is not in the top ten. Not only normalizing the data but using a different metric can also results in different outcomes. Using the average number of packets sent per IP address to classify countries results in a similar ordering of countries as using the number of packets sent of the number of IP addresses. However, China is not present in the list even though most packets are sent from it and harbours the most IP addresses used for scanning. The different results obtained by raw data, normalization, and the use of different metrics show that it is possible to manipulate the outcome of data processing in order to achieve a desired conclusion. Therefore different metrics, and normalizations along with the raw numbers should always be presented.

NUMBER OF TCP SYN PACKETS

The number of packets sent per country varies widely. It becomes clear that some countries send more packets than others. More interesting, plotting the number of probes sent per country shows that the number of packets sent by scanners in a country, has a similar distribution compared to the number of IPv4 addresses used to send probes from per country

Table 7.1 shows the top 10 sending countries in terms of packets. The table also shows that not all countries are equal in the number of packets sent, the top ten sending countries are responsible for 85.73% of all recorded traffic. The table also reveals that the countries sending the most packets are not the same as the countries using the most IPv4 to perform the scans. However some countries are present in both tables, such as China and Russia.

Merging the number of IPv4 and number of packets sent we can look if there is a relation between it. Table 7.1 shows that there are some countries present some columns. However for the other countries in the dataset no relation between the average number of packets and number of IPv4 per country can be detected.

One may only conclude that the country from which most scanning originates greatly depends on the chosen metric. For nearly every chosen metric the top ten countries from which most scanning activity is observed varies. Nevertheless China can be seen in absolute numbers as generating most scanning activity, 45% of the recorded traffic and 37% of the source addresses having sent probes.

USED SCANNING TOOLS

As mentioned in previous sections, several scanning tools are identifiable. An analysis of the used by geographical locations could reveal if some tools are preferred in some regions of the world. For the analysis, scanners using ZMap like software are chosen, the source addresses sending probes using the tool are mapped onto the world map.

The plot 7.4 of the world map shows the number of addresses (in dot size) using different flavours of ZMap like tools according to the fixed IP identification value (in colors). The world map shows that the use of different types of the tools are spread over the world map. The widespread usage of different tools suggest that the tools are well known by scanners or that scanners using these tools use several source addresses associated with different countries. The latter is more interesting as it suggests scanners using a wide network of varying IP addresses to perform scans.

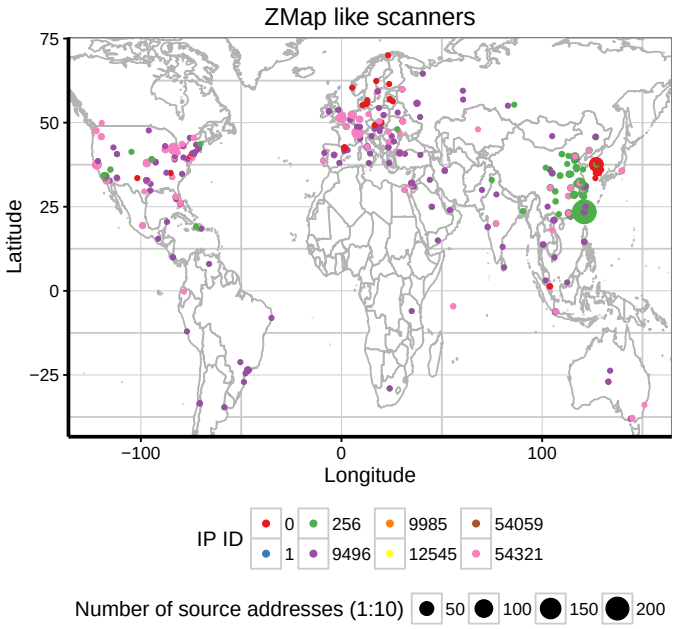


Figure 7.4: Geographical location of IPv4 addresses having sent ZMap like TCP SYN packets

Although the identified ZMap like tools are used globally, some are more frequently used in some geographical locations. Analysis of the data shows that ZMap like tool fixing the IP identification value to 256 is predominately used in Asia. More so, scanners probing from addresses located in Taiwan prefer the usage of the ZMap like tool having fixed the IP identification field to 256. Over 2000 source addresses send packets with the fixed IP identification field whereas only 50 use other fixed values. Similar behaviour is noticed in South Korea where the tool of choice is ZMap like, fixing the IP identification field to 0.

WHICH AUTONOMOUS SYSTEMS ARE MORE ACTIVE

In the previous section we looked at scanners from a country perspective. A similar approach can be made using the autonomous system (AS) numbers. These numbers are associated with the IPs and on a higher level with IPv4 block assigners such as LIRs and RIRs. The AS numbers are used by the border gateway protocol (BGP) to exchange information about the location of IPv4 address to facilitate the routing of packets through the Internet.

AS numbers are somewhat related to countries, even though they are not suited to provide the location of an IP address. Figure 7.5b shows the number of IPv4 addresses per AS number present in the dataset. The figure shows 6 groups, which can be explained by the allocation of AS numbers. The IANA has since 1 December 2006 moved from 16 bit AS numbers to 32 bit AS numbers to increase the number of AS numbers it can allocate. During the expansion of AS numbers the old 32 bit AS number were kept, depicted by the cluster between 0 and 65,536 in figure 7.5b. The allocation of new AS numbers is performed in a non consecutive manner [63] resulting in the five AS number clusters above 65,536 shown in figure 7.5b.

As shown in figure 7.5b, there are some AS which occur more often, compared to others. Table 7.2 shows the top autonomous systems, related to IPv4 having scanned the TU Delft. The results bear resemblance with those presented in the country section in table 7.1.

The number of packets per AS is shown by figure 7.5a. This also illustrates that some AS are more used than others and points out similarities with the data of the countries presented in table 7.1.

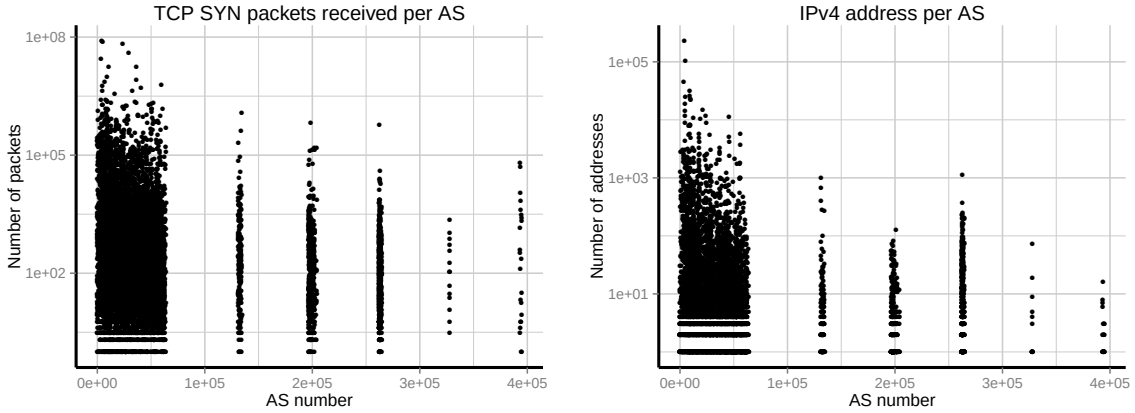
Table 7.3 is similar to table 7.5a. Suggesting that these AS are not only responsible for most of the traffic but also are hosting the majority of IP addresses.

AS Number	AS Name	Number of IPv4
4134	CHINANET-BACKBONE No.31,Jin-rong Street,CN	230 334 (20.28%)
4837	CHINA169-BACKBONE CNCGROUP China169 Backbone,CN	105 877 (9.32%)
3462	HINET Data Communication Business Group,TW	45 060 (3.96%)
8926	MOLDTELECOM-AS Moldtelecom SA,MD	31 321 (2.75%)
9121	TTNET Turk Telekomunikasyon Anonim Sirketi,TR	26 015 (2.29%)
Total	-	438 607 (38.6%)

Table 7.2: Top 10 AS related to the IPv4 having sent TCP SYN packets

AS Number	AS Name	TCP SYN packets
4134	CHINANET-BACKBONE No.31	80 574 808 (14.41%)
4837	CHINA169-BACKBONE CNCGROUP China169 Backbone	76163819 (13.62%)
23650	CHINANET-JS-AS-AP jiangsu province backbone	67 540 032 (12.08%)
29073	QUASINETWORKS Quasi Networks LTD.	41 010 608 (7.33%)
3462	HINET Data Communication Business Group	28 750 236 (2.29%)
Total	-	294 039 503 (52.61%)

Table 7.3: Top 5 AS sent TCP SYN packets



(a) Number of TCP SYN packets sent per AS number

(b) Number of IPv4 addresses sent per AS number

Figure 7.5: Number of packets and IPv4 addresses per AS

The analysis of the data using AS numbers as basis, reflect the findings of the work in previous subsection performed on countries. Autonomous systems which are the most actively used by scanners in terms of packets sent and addresses used reside in China. However when correcting for several factors such as number of addresses different autonomous systems can be marked as most actively used.

8

TIME ANALYSIS

This section presents the time analysis of scanners. This section will illustrate if there are differences between scanners in a time manner. Related work distinguishes between slow and fast scanners, time analysis might reveal characteristics of these classes of scanners. Such characteristics include whether the attackers perform attacks over a number of days or that they prefer to scan for a short period of time.

DAILY ANALYSIS

In order to get a better understanding of the data it is imperative to look at it from a time perspective. Time is a valuable aspect in scanning as intrusion detection systems use time as a identification factor, additional time indicates if the scanners tend to be active for long or short time periods. Furthermore a time analysis may reveal if the network is constantly scanned with the same intensity.

Figure 8.1 shows the normalized number of scanners and received packets per day. First, the curves are not flat, indicating the network is not subject to the same amount of scanning traffic the entire time. It also suggests that using a dataset with a smaller time interval is not desirable as it might not represent scanning activity. If a dataset is collected in a period of high activity such as monitored around April the 20th a biased view on regular scanning activity might form. Furthermore figure 8.1 demonstrates the correlation between the number of scanners and the number of packets received.

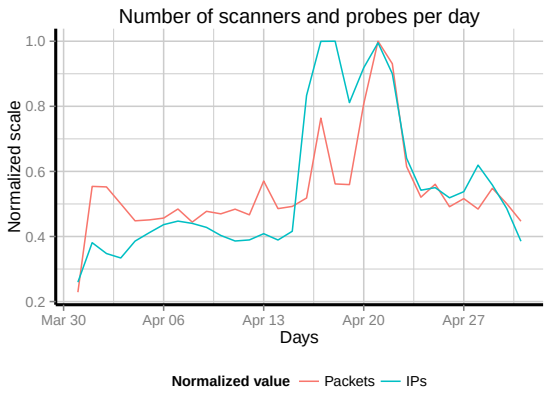


Figure 8.1: Normalized number of scanners and received packets per day.

Figure 8.1 shows some anomalies, caused by a large increase in number of packets received between the 16th and 23rd of April. First, around the 16th and 19th, an increase of IP addresses scanning the network accompanied by an increase in packets is observed. The increase in source addresses and packets suggests that more scanners are scanning the network, each sending an average amount of packets. However between the 20th and 22nd another peak is visible, also in terms of scanning IP addresses and received probes. Compared to the first peak, the number of packets received is much higher whereas the number of IP addresses is the same. The observed ratio between number of packets and addresses seems abnormal, an analysis of the data shows that there is a single scanner which at that time is responsible for 40% of all the collected data. The scanner responsible for that peak in number of packets scanned for two days.

The discovery of a scanner, scanning for two days at a high rate, raises the question on the number of days each scanners scans and if the number of days scanned are equal. Several distinct types of behaviour conceivable in terms of time scanned. As scanner may perform a hit and run, which translates to one continuous

scan during a short period of time. Another hypothesis would be that scanners are scanning continuously for a long period of time, which could be as long as the collection period. Finally, a third behaviour would be a somewhat random daily or even weekly scanning pattern. Scanners scanning in an erratic pattern, performing scans of different lengths (in terms of time) separated by different length time intervals.

Figure 8.2 shows that the number of days scanned is not uniformly distributed. It seems that the majority of the scanners scan one day whereas the minority are sending probes over the course of multiple days. Furthermore figure shows 8.2 a peak at 31 days, indicating scanners being active for the entire recording time. 70.20% of the attackers scan for one day, 16.33 % for two days and 5.09% for three days and less than one percent for the full length of the recording period.

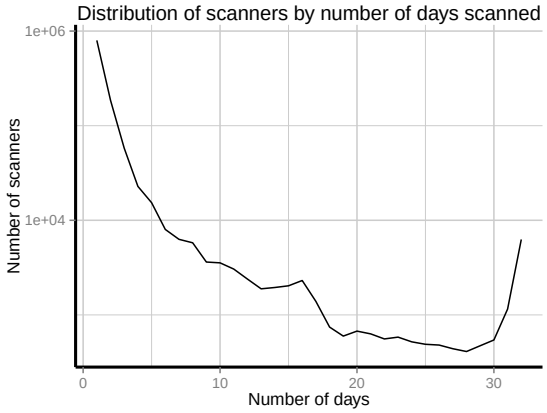


Figure 8.2: Distribution of number of scanners by number of days scanned

Looking at the number of days scanned versus the number of packets sent one can see that the majority of attackers send probes for one day sending less than 100 packets. Another phenomenon is that scanners send certain number of packets per day over others, namely multiples of 256. The reception of probes in multiples of 256 can be explained by the scanner scanning a specific subnet, and by the nature of defining IP ranges which is done in multiples of 256.

Furthermore, some scanners send the same amount of probes on different days to different subnets. The scanning of different subnets over the period of several days may indicate that the scanner does not have the resources for sending a large amount of traffic thereby spreading the number of packets sent over several days. An alternative is that the attacker deliberately lowers his probing speed to avoid being detected by intrusion detection systems using a packets threshold. In order to further investigate behaviours of scanners on a time scale, a smaller time window must be used to represent the number of probes sent.

HOURLY ANALYSIS

Beside the daily activity, the hourly activity of scanner can also be analysed, as seen in the daily scanning activity there is significant variance on number of packets received and source addresses used for scanning per day. A finer time scale might reveal if scanners have a preference for a certain hours of the day. One may argue that attacks are more prevalent during the day. For DDoS attacks alike, which try to render a service unavailable for the users. It is self explanatory that the attack should be performed during the day, or hours when users are more likely going to use the targeted service. Therefore the traffic/attacks should be more prevalent during some hours of the day than others.

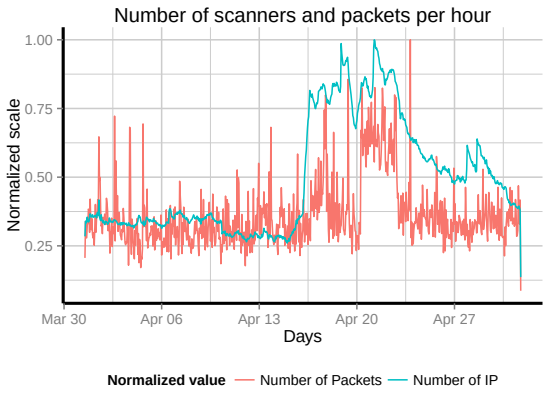


Figure 8.3: Normalized number of scanners and received packets per day.

The question remains whether or not scanners prefer some hours over others for conducting scans. Differences in hourly scan traffic might give an insight into the targets of the scanners. If the scanners are targeting for a broad audience, a difference in daytime and night time traffic should be visible, because the targets are more likely to be online during the day then during the night, or other specific hours of the day. If no difference is noticed in scanning times, the targets of the scanners are likely machines continuously connected to the internet or the scanner has no specific target.

When looking at figure 8.3 we can see a clear distinction between the number of IP addresses and the number of packets received. During the first half of the month the number of IP addresses is stable. However the number of packets strongly fluctuates. Looking at the data we can see that the spikes in number of packets shown in figure 8.3 is caused by individual IP addresses which perform short but packet intensive scans. During such a scan they send almost the same amount of data that is generated by the other scanners. Some scanners even perform multiple bursts days apart from each other.

Another interesting phenomenon, shown in figure 8.3 around the 20th of April, is an increase in both the number of packets and the number of IP addresses. The sudden increase in source addresses observed around the 20th of April suggests that a large number of IP addresses working together are performing a scan. Analysing the data reveals that the number of source addresses sending probes more than doubles from 12,000 to 33,000 in less than a day. Further research based on the source addresses shows large adjacent IP blocks performing scans at the same hours.

The hourly plot does not reveal much other than scanning activity being erratic, as there are too many fluctuation caused by burst of scans. The fluctuations in number of packets captured confirms the existence of scanners sending high amount of traffic in a short period of time. The activity of burst scanners can be identified on figure 8.3 by an increase in the number of packets accompanied with no increase in monitored source addresses.

It could also be that the scanners only have access to their equipment during a certain period of the day. Time restricted scanning abilities would show when the hours scanned per scanner are corrected for the UTC offset, meaning that the scans are looked at from a local time perspective.

Correcting the data for local UTC time does not reveal much more information. It reveals that scanners do not base their activities on their local time. No increase in packets sent, nor any correlation with the number of scans can be seen in the data. Affirming that not only the number of scanners is constant throughout the hours of the day, but also the generated traffic.

Although the scanners do not seem to prefer a particular time of day, some addresses belonging to the same subnet, even adjacent addresses, send the same amount of packets during the same hours. Using this information an efforts done to perform clustering based on the number of packets sent during a certain hour of the entire collection period.

The used clustering method is k-means, as described by Hartigan [64], however no suitable output is generated. K-means requires an expected number of clusters as input in order to form the clusters. A method of determining the numbers of clusters needed is the elbow methods, which consists of running the k-means algorithm for a range of clusters and calculate the sum of square errors. The appropriate number of clusters for the data is determined by selecting the number for which the curve of sum of squared errors flattens. Calculating the sum of square errors for a range of 20 clusters reveals that the elbow of the curve of square sum of errors lies between two and three clusters. Analysis of the output of k-mean algorithm on the data for 2 or 3 clusters does not yield in clusters revealing time wise similarities between scanning source addresses. The lack of information given by the division in few clusters is due to the widely varying scanning activity in terms of scan times.

The small amount of clusters can be attributed to the high numbers of scanners only sending probes for one hour. K-means is an iterative algorithm, recalculating the centre of the clusters at each iteration, the algorithm stops when the position of the old centre and the newly calculated centre converge. Therefore, a highly varied dataset, scanners can scan a combination of 720 different hours during the collection period, will not fare well as the centres of the clusters will quickly converge thus reducing the number of clusters detectable.

Another option would be to ignore the elbow method and increase the number of clusters. However, the clustering problem solved by the k-means algorithm belongs to the non-deterministic polynomial time (NP) class. The NP class of problems has the characteristic of not being solvable in polynomial time, which translates into NP problems taking exponentially more time when the input is increased. Therefore, increasing the number of clusters beyond 100 is not feasible as the run time of the k-mean algorithm would become to long.

A second attempt at differentiating possible scanners scanning the same period of time is done through visualization. The visualization of the dataset on a time scale is challenging as there are 720 hours recorded which results in a dataset with 720 dimensions. Visualizing 720 dimensions is practically impossible, therefore the number of dimensions should be reduced to be able to view the data in a 2d plot. Using t-sne,

developed by Maaten and Hinton [65] the number of dimentions of the dataset is reduced and the result is plotted in two dimensions. The t-sne algorithm uses the number of packets sent per hour as an input vector for each source address. The distance between between source addresses is calculated using the time vector. The calculated distance is converted to a probability indicating whether the two addresses are likely to be neighbours in a plot. The conversion of distance to probability results in attackers scanning the same hours with the same amount of packets to have a high probability of being neighbours, resulting in them being plotted near each other.

The algorithm plots the majority of IP addresses within a cloud at the centre of the plot. However, some addresses form identifiable groups which are located outside of the centre cloud. An inspection of several groups outside of the cloud reveals that the source addresses indeed scan during the same time period and send the same amount of packets. More so, part of the grouped addresses are part of the same subnet, even adjacent. Additionally some identified groups share probe characteristics such as fixed IP identification number and sequence numbers.

The time analysis based on the number of hours scanned has reveals that scanners do not target the network at specific time of day. Also source addresses attack during the same hours, and send the same amount of packets. Additionally some of these source addresses use the same header values and belong to the same subnet. The observation of neighbouring addresses having the same behaviour suggests that these addresses are working together. The potential cooperation of addresses is discovered using a time metric on the data. However, using hours as a scale has limitations as it is a hard set threshold, therefore in the next subsection a new time scale is presented using a variable threshold.

SCAN TIME ANALYSIS

Even though the daily and hourly scan activity has provided valuable information, the time scale of the scan duration is fixed. The days are defined by the days counted on the server, and the hours are also fixed time intervals. Scanners having sent packets short before midnight and afterwords will be recorded as having scanned two days. Likewise a scanner having scanned continuously for 20 minutes, scanning the last ten minutes of the first hours and the first 10 minutes of the consecutive one will be recorded as having scanned two hours. Metrics based on fixed time periods, although providing interesting information might not accurately depict the reality.

Therefore, a non fixed time interval should be chosen to depict the real scan times of attackers. A simple approach would be to set a time-out between arrival times of packets of the same scanner. Implementing a time-out value would allow for a clear definition of a scan. In order to investigate whether such a time-out would work, one needs to look at the arrival times between packets.

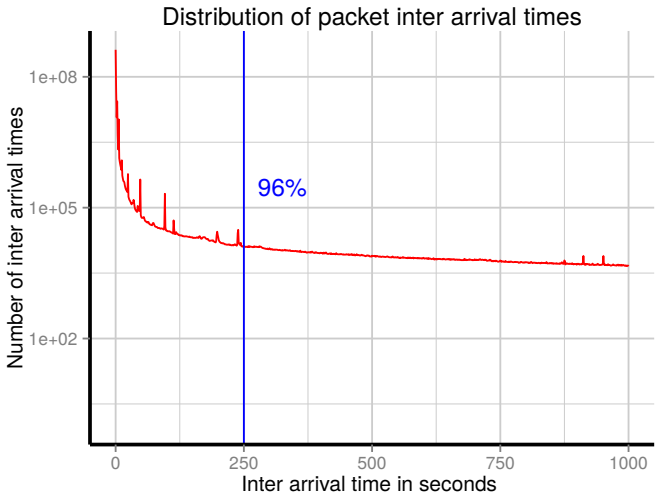


Figure 8.4: Distribution of time intervals between packets

Figure 8.4 shows the distribution of packets having inter arrival times in seconds, and a blue line showing 96% of the data. Figure 8.4 confirms that the use of a hard set time out would work for a majority (96%) of the received traffic. A value of 250 seconds for the time-out would classify a large majority of the packets received as one scan. However we can see that using a method with a fixed time-out value would also discard the rest of consecutive packets as not being part of a scan. Scanners using a low scanning rate, of more than 250 seconds in between per packets, would not be classified as continuous scans. Such scanners are not inconceivable, as scanner detection software uses hard set limits, therefore forcing the scanner to use a lower packet rate in order to remain undetected. Presumably scanner lowering their probe

rate to over 250 seconds would also not be detected, using a hard set limit.

The naive approach would work for a majority of scans, however slow scanners should also be detected as performing slow scans. Hence a new method should be devised. The proposed approach uses not only a hard time out value but also a calculated one which varies depending on the scanning speed of the attacker. For each packet received of the scanner the time in between packets is calculated, then the average and variance are calculated and updated. If the new time interval lies within two times the standard deviation, the packet received is still part of the scan. The proposed method is based on the normal distribution. If a scanner is scanning at a constant speed, the received time interval should form a Gaussian curve with the centre at the mean of the time intervals and with 90% of the time interval within two times the standard deviation. Therefore if the variable limit is set at two times the standard deviation plus the mean of the collected packets one can accurately determine if a received packet is still part of a scan.

The results using the proposed method are shown in figure 8.5 where the number of scans is depicted against the scan duration. The proposed method reveals that 26.87% of scans are less than 1,250 seconds or 20 minutes long. Not only do the scanners prefer short scanning times the majority also scans less than 100 times, with most of them only scanning once. The maximum number of scans where performed by what now seems to be a Juniper switch 218.17.147.45, which has performed 1,347 scans. The switch could have been compromised and used as the scanning device using the routed traffic through it to hide the scanning activity. Or, more likely, the switch also uses network address translation (NAT) and the scanning devices are behind the switch. As expected we can see two peaks around 300 seconds which is the hard set time-out value. Also 3% and 1.5% of the scans have a duration of respectively 235 and 285 seconds.

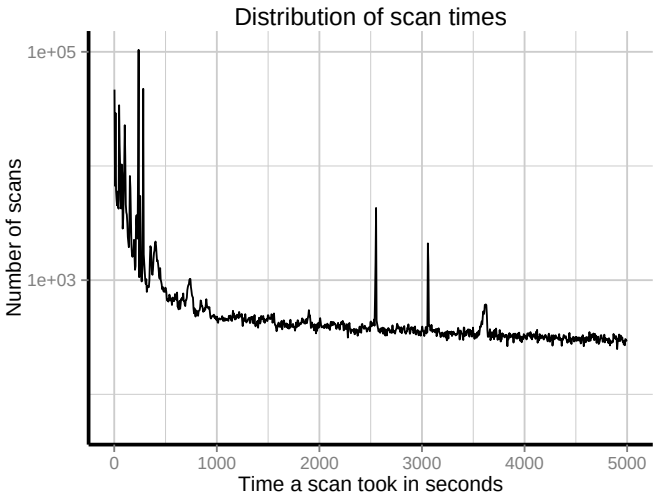


Figure 8.5: Distribution of scan duration

Manual inspection of scan with a duration of 235 seconds reveals that these scans come from addresses within the same subnet and even adjacent addresses. The source addresses from which the scans originate not only scan for the same duration, they send the same amount of probes. More so, the time in between reception of probes of different source addresses is also equal. The similarities in scan properties, time, number of probes and inter probe arrival time, suggest that the scanners attacking from these source addresses are using the same tools.

The rest of the duration of scans are, with three exceptions, equally distributed between 20 minutes and the rest of the collection period. Three peaks shown on figure 8.5 between 2,500 and 3,750 suggests that some scanners have similar behaviour. Manual inspection reveals the scanners sending the same amount of packets and some source addresses having the same probe sending rate, having the same time period between sending two packets.

Detecting similarities in scan duration and speed for multiple source addresses raises the question whether time based behaviours can be linked to used scanning tools. To answer the question an analysis is performed on source addresses using ZMap like scanning tools, having the IP identification field fixed to 0, 256, and 54321. It seems that the three different tools are used to perform different types of scans. The scanners using the tool with a fixed value of 0, are slow scanners compare to scanners using other aforementioned tools, 88% of the source addresses send less than one packet every 10 seconds. Comparing to the original ZMap tool which the only 0.89% of the scans has a speed of less than 1 packet every ten seconds. Contrary the ZMap like tool fixing the IP identification number to 256 seems to be preferred by fast scanning attackers, 98% of

the scans send more than 10 packets per second whereas 98% of the scanners using the original ZMap send between 1 and 4 probes per second. It seems that the attackers chose different tools for different scanning techniques. Compared to the original ZMap software other similar tools, using 256 and 0 for the IP identification field, seems to be only used for fast or slow scans. This would imply that attackers have developed tools for specific tasks.

The proposed method to determine the duration of scans reveals that 25% of the attackers send probes for less than 20 minutes and that the rest of attack durations is equally distributed, revealing that a large amount of scanners prefer short scan duration. The method also allows for the comparison of scanning behaviour as it provide the actual attack speed and accurate time intervals between scans. The proposed methods also allows to compare how identifiable tools are used to probe the monitored range. Difference in the usage of tools, slow or fast scan, is noted based on the scanning speeds derived the number of packets sent and the duration of an attack.

9

SCANNER BEHAVIOURS

This section introduces the different types of scanners found using the analysis methods presented in previous sections. Various other papers have identified a wide range of scanners, namely Bhuyan et al. [17] have devised a classification of port scanners. First they distinguish between scanners using single or multiple source IP addresses to perform scans. In addition the authors also make the distinction between scanners sending probes to a single or multiple destination addresses. They also combine their results with scanning for single, multiple or all destination ports on the targeted machines. These results in an already elaborate classification of scanners, the vertical scanner which scans for some or all ports on a single target. The horizontal scanner scans a single port on multiple targets. The strobe scanner, also scans multiple ports, however it does so on multiple targets. Finally the block scanner which scans all ports on multiple targets.

The proposed classification covers all the possible scanning behaviours for a single sourced scanner considering targeted ports and IP addresses. This classification can also be projected on scanners using multiple source addresses as it also reflects the scanned destination IP addresses and ports. However the classification only shows part of the behaviour of the scanners, as it only reveals the targeted ports IP addresses and does not go further into the dynamics of the performed scans. Therefore in this section attackers will be analysed in order to uncover whether additional characteristics can be found. In order to adhere to the common classification of port scanners, this chapter contains two parts, distinguishing between single and multi source scanners.

SINGLE SOURCE SCANNERS

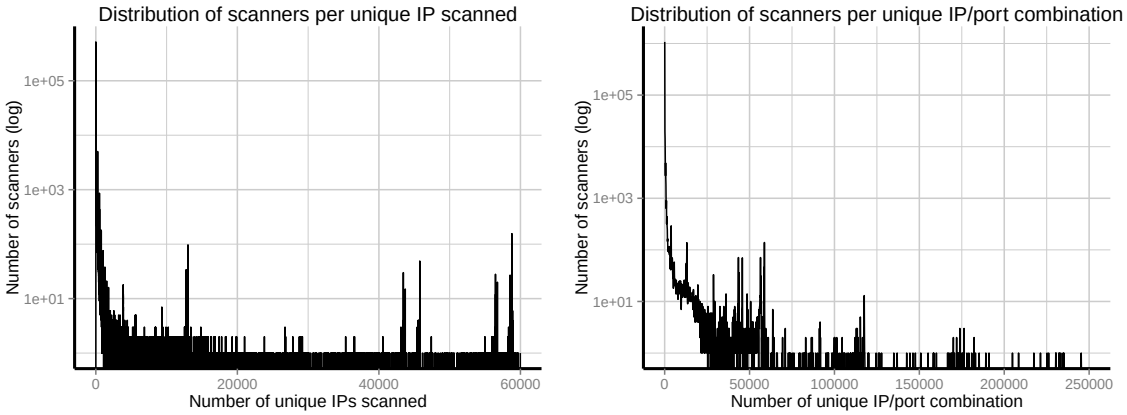
Single source scanners, as defined previously, only use one IP address. An interesting perspective is to look at the repartition of these scanners in terms of the previously mentioned classification. The distinction of classes can be done by examining the number of address and port scanned by different attackers, providing an insight into the number of scanners belonging to each class of scanners (horizontal, vertical, strobe).

GENERAL OVERVIEW

First, from the data can be deduced that the majority of the scanners only scan for a single IP address. This is reflected in figure 9.1a showing the distribution of scanners in relation to the number of IP addresses scanned. The results form the figure are confirmed by the cumulative distribution function showing that 99% of the scanners only send packets to one destination address. This would indicate that the majority of the scanners are vertical scanners, or are scanning multiple ports on a single IP.

In order to confirm this the number of unique ports scanned per scanner needs to be investigated. Plotting the number of ports scanned per scanner does not reveal much, as the only two scanners scan more than 5000 ports. The majority of the scanners, 99%, only probe one port. This, in combination with previous findings, would suggest that most of the scanners only target one address on a specific port. Therefore, the number of unique IP and port combinations scanned per attackers should be analysed.

Figure 9.1b shows that the majority of the scanners, more than 99%,indeed scan only one destination IP port combination. This can directly be linked back to the fact that most of the scanners only send one packet. However there are, according to figure 9.1b also scanners having scanned a high amount of unique IP and destination port combinations.



(a) Distribution of scanners according to unique destination IP addresses scanned

(b) Distribution of scanners according to unique destination IP addresses scanned

Figure 9.1: Unique addresses and address/port scanned per source address

The behaviour of scanning unique destination ports and addresses leads to a new class of attackers, named unique scanners, resembling the strobe scanners, scanning multiple port on multiple addresses. These scanners scan one port on one IP address and then another one on another IP address, never sending probes to the same port or address twice. This is shown in figure 9.2, the number of unique IP addresses are on the x axis and the number of unique addresses on the y scale. There is is a distinct line visible of scanners, highlighted by the red line, which is the result of them having scanned the same amount of unique IP addresses as unique ports. This demonstrates that there are multiple scanners which not only scan several IP addresses and ports, such as strobe scanners, but they never scan the same port nor IP address twice.

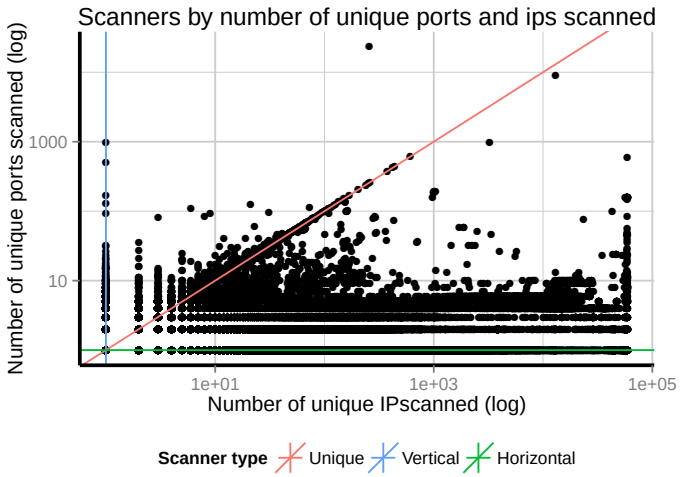


Figure 9.2: Plot of scanners showing the umber of unique IP and unique ports scanned

Another behavioural aspect is the correlation between used tools and scanned ports. Research conducted at the TU Delft by Doerr et al. [66] shows that some tools are used more often to scan specific ports. The results are based on recording the number of packets sent by addresses using identified tools as explained in previous section (NMap, ZMap, Unicornscan and Masscan). The research shows that scanners prefer scanning for SSH services using Unicornscan and in lesser extent ZMap, whereas NMap and Masscan are seldom used to scan the port associated with SSH.

The behavioural aspect of using different tools to scan different ports is also noticed in other tools detected during the research period of the thesis. In previous sections ZMap like scanning software has been introduced using fixed IP identification numbers. These tools, as shown by table 9.1, also seem to be utilized to scan specific ports, other than port probed by scanners using ZMap. Specifically the tool using IP identification number 256 is mainly used to probe port 0. Port 0 is defined by TCP not to be used and therefore has not specified the correct response to packets arriving to that port. The responses depend on the implementation of the operating system (OS) and can therefore be used for OS fingerprinting as described by Jones [67].

This section has confirmed that the known scanning classes are still prevalent amongst the scanners. In addition to the already known categorization, another behaviour has emerged: some scanners target different ports on different addresses, never targeting the same port or address twice.

	IP id. = 0		IP id. = 256		IP id. = 54321	
Top 5	Port no.	Packets	Port no.	Packets	Port no.	Packets
1	0	40.83%	1433	17.11%	22	32.69%
2	1723	21.90%	9200	8.82%	443	11.63%
3	23	14.77%	37564	6.94%	80	8.09%
4	8080	2.03%	3306	6.57%	8080	4.76%
5	8070	1.35%	22	6.55%	21	3.44%

Table 9.1: Percentage of packets received on destination ports classified by used tools

Also has been revealed that scanners use different tools to scan different ports. Previous section has revealed that some tools are used more abundantly in some regions of the world than other 7.4. Especially scanning tool using a fixed IP identification number of 0 is mainly used in Korea. Combining the two results suggests that scanning activity related to OS fingerprinting is mainly conducted in Korea.

Even tough those different classes of scanners enable their categorization, it does not reveal the dynamic of scans, it only allows classifying them by targeted addresses and ports. Also the trends of using tools for specific target ports does not reveal individual scanning behaviour. Therefore in the next section a closer look will be performed on some scanners in order to uncover behavioural properties of scanners.

PARTICULAR BEHAVIOUR

This subsection analyses some scans performed during the period of data collection.

The information gathered in previous sections shows that the detection of single source scanners can easily be achieved. Detection of such a scanner can be done by looking for abnormal behaviour, which is present in each of the proposed analysis. Such behaviour can be detected by analysing the traffic sent from an address using criteria mentioned in previous chapters such as scanning time, number of packets, or packet properties such as fixed header values. When combining all these factors, an address can be singled out which is most likely to be a single host scanner as it is the only address adhering to the selected behaviour. Although combining the different characteristics of an address to increase the certainty of having a single source scanner is sensible, it is not always necessary.

Table 6.1 shows the number of IP addresses sending packets with static IP identification field, which is a trait of ZMap like scanning tools. The table shows that four IP identification fields are associated with one address each, indicating that there are four single source scanners each using a specific IP identification number. The use of a static identification number by scanners is not unusual. However, the other fixed numbers such as 256 and 5,4321 are used by multiple scanners, these four are not indicating that these are indeed single source scanners. These single source scanners have either made their own software which has ZMap like fingerprint or have simply altered the ZMap code in order to change the IP identification number.

One of these scanners is 180.97.192.16 which uses the IP identification number of 9,985 for each of the 16,279 packets sent to our network. An IP lookup informs us that the server is located in China, which is not surprising as it harbours the majority of addresses used by attackers. Further investigation shows that the server hosts a website, however the access is restricted to an IP white list. The scanners seems to be a strobe scanner, scanning four ports on multiple destination machines. Looking at the scans from a time perspective reveals that the scanner has performed 12 different scans as defined in previous section. The majority of these scans are performed during the end of the collection period. However no uniformity can be seen in the speed, number of packets nor duration of the scans.

Analysing the targeted IP addresses unveils that the scanner sent probes to 13,447 unique addresses in the collection range. As mentioned, the scanner is interested in four ports as shown in table 9.2, revealing not only the scanned ports but also the number of packets and addresses scanned per port. This table shows a clear division between the targeted ports, the number of packets sent, and addresses probed on port 80 is twice as high as the other ports. It is clear that the emphasis of the scans performed lies on port 80, and therefore one is inclined to say that the scanner himself is focused on port 80.

This deduction can be challenged by arguing that the scan was recorded at the end of the collection period and therefore might give a biased picture of the scanner. It is possible that the scanner first scanned for port 80 during the recording period and then shifted his attention to other ports at the end of this period excluding his scans for other ports from the dataset. An analysis of the raw scanning data shows that this is not the case as

all the ports are being actively scanned at the same time, therefore eliminating this theory. In addition to the seemingly random behaviour in terms of port succession, the data reveals another remarkable characteristic of the scanners behaviour. The scanner scans his targets in an orderly fashion, incrementing the destination address with one each time a new probe is sent towards the same ports.

This scanner shows that there are tools being used with particular behaviours when it comes to the order in which the destination addresses are probed. As mentioned before this scanner is classified as using a ZMap like scanning tool, due to the fixed IP identification number. However the scanning order of IP addresses leads to believe that this scanner uses a heavily modified version of ZMap or a personally written piece of code because ZMap scans the targets in a random fashion. This is resulting in a class of scanners modifying or using their own scanning tools, contrary to the other scanners using readily available tools.

The aspect of scanners using unknown tools is not the only behaviour exposed by this scanner, the ratio of the traffic port wise also reveals a behavioural characteristic. As mentioned before, this is a strobe type of scanner, however this attacker also possesses the capability of managing his resources. This can be concluded from the fact that the ratios of targeted ports are disproportionate, showing the capability of the scanner to intensify the probing of some ports over others.

Another single source scanner is 146.185.239.104 using another non-standard fixed IP identification field of 5,049. The IP address is the property of a server renting company name Cubehost located in Russia. The scanner can be classified as a continuous scanner, due to the longevity of his scan. During the entire data collection period, the scanner performed 2 continuous scans separated by 6 minutes spanning the entire collection period. During these scans, the attacker sent a steady stream of 401,649 packets with an average of 6.72 seconds in between probes.

Destination port	Packets	Destination IPs
80	6,408	6,081
3,128	3,380	3,370
8,080	3,261	3,252
9,064	3,230	3,217

Table 9.2: Port scanned by 180.97.192.16

Similar to the previous scanner, this attacker scans for a small number of ports, also classifying him as a strobe scanner. Table 9.3 shows that this scanner also prefers port 80 over the other scanned ports. An analysis based on time and ports, such as performed on the previous scanner confirms that the attacker focuses his probing on port 80. The focus on a specific port by this scanner and by the previous one confirms that scanners have the possibility to manage their resource in order to achieve a higher probing rate for some ports.

Although this scanner shares properties with the previous attacker, they also have their differences. Contrary to the previous scanner, this one does not scan the destination IP addresses in an ordered fashion. Also, this attacker uses a different identification field for all the generated traffic, raising the suspicion that the scanner is using a different tool. This suspicion is confirmed by the usage of two, fixed, sequence numbers during the two scans, which is not the case for the previous scanner, which has a random sequence number. More so this scanner only sent traffic from one source port whereas the previous scanner sends traffic from different source ports. These sheer differences leads to consider that both scanners used different non-known scanning software.

The radically different behaviours in terms of header values, confirm the suspicion of scanners constructing their own software. It also shows that there is a different degree of complexity in the written software. One can argue that the usage of random targets is more sophisticated than simple incrementing the next target because the likelihood of scanning two target parts of the same network is much smaller thereby reducing the chance of detection.

The third ZMap like scanner uses IP identification number of 12,545 and originates from 112.221.251.221. The address is linked to a server belonging to LG DACOM located in Korea. The scanner is distinctive because he uses a static source port and sequence number, per destination port scanned (21 and 179). Contrary to previous scanners, he has performed two different continuous scans for each targeted port. More so, the scanners use an interesting scanning pattern not seen in the two previously analysed scanners.

Source port	Packets	Destination IPs
80	268,099	56,408
3,128	66,728	27,144
8,080	66,822	27,100

Table 9.3: Port scanned by 146.185.239.104

During the scans the targeted IP addresses are also ordered. However contrary to the previous example,

the consecutively probed IP addresses are not incremented by one but by 255. Resulting in packets addresses to XXX.XXX.123.YYY being followed by XXX.XXX.124.YYY, where YYY is equal. More so, the scanner scans both the 131.180.0.0 and the 130.161.0.0/16 range in this manner. In addition the packets received for the same IP addresses in the different ranges of the university, ex: 131.180.ABC.XXX and 130.161.ABC.XXX, are always simultaneously received. There are some exceptions to be noticed in the data when it comes to this pattern of probing same addresses in different ranges simultaneously, however they can be explained by the allocation of IP addresses by the TU Delft. As disclosed in previous section some parts of the TU Delft are assigned more than others, this explains why packets for the same addresses in different ranges are not always collected in the data. This behaviour leads to the notion that the scanner probes chunk of the Internet simultaneously as both networks receive packets destined at the same address at the same time. Further analysis of the data shows that the attacker sends packets with an interval of 3 seconds. When combining this with the assumption that the scanner is scanning the entire internet an estimate of the scanning speed can be made. Based on previous data, the attacker scans 255*255 IP addresses per three seconds if he would be scanning the entire Internet. If the assumption is correct the scanner sends 21,675 probes per seconds. A crude analysis of the TCP SYN packets performed by Miessler [68] shows that the size of such packet type is 60 bytes. Combining these two values results in a scanning speed of 1.3Mb/s

.

These three scanners indicate that not only do scanners have the possibility to manage their resource but that they also use different tools. In addition the scanning pattern from these scanners are widely different in terms of headers values and target order. However there are some similarities between two of the scanners, namely the targeted ports. Both the first and second scanners target the same ports, from which two are in the top ten of targeted TCP ports 5.3a. This can indicate possible similarities in the intentions of the scanners as the probed ports can be related to known malware such as My Doom described by Srinivas [69]. However this remains an intuition needing additional analysis.

The first three scanners analysed stand out by the fixed IP identification number, whereas this one does by the number of distinct scans performed. The attacker uses the IP address of 218.17.147.45 which is linked to a Juniper gateway located in China. From this source address, 1,347 distinct scans are recorded during the collection period, sending in total 59,748 packets which classifies this scanner as slow. All traffic generated by this scanner is destined for port 80, classifying this scanner as a horizontal. More interesting is that this scanner does not seem to be using an elaborate tool unlike the previous ones.

First the scanner cannot be associated with any of the known scanning tools. This can be concluded by the analysis performed in previous chapter identifying which addresses can be associated with known scanners. However it can also be done through simple deduction due to the scanner sending three packets to each destination address. When the scanner sends two packets to the same address, the sequence number and source port stay the same, additionally the IP identification field remains unchanged. This eliminates Masscan and NMap for respectively changing the IP identification field value and keeping the same sequence number. Unicorns scan can be eliminated by calculating a few session keys, which results in a different key for each destination address. This would imply the scanner having to restart Unicornscan every time he probes a new destination address within the range of the TU. Finally it is obvious that ZMap is not used due to the varying IP identification field.

This leaves us to believe that the scanner is using a custom tool, which cannot be associated with known scanning tools. The retransmission of three packets to the same destination address reveals more information. There is a high probability that the scanner is using the standard TCP stack. This can be confirmed by the time in between retransmission of packets, which doubles after each failed transmission. From the data can be seen that the time between the first and second packet to the same destination address is three seconds. The time between the second and third is six seconds. This adheres to the design of TCP which doubles the time in between packets when no acknowledgement is given. In addition the sequence number for the retransmission of packets remains the same. Furthermore the retransmissions are ordered, meaning that the scanner first sends three retries before probing another target. These observations lead to the conclusion that the attacker is using a simple tool using the standard TCP stack and is not able or not willing to craft packets.

Nevertheless, the scanner uses a random pattern to scan the range of the TU Delft, revealing that although the attacker is not using a sophisticated tool he tries to stay undetected by intrusion detection software. Remarkably the attacker is using a switch to scan the internet, and therefore may not have the required privileges to craft packets or no scanning software is available for the platform. The scanner is an example that attackers do not only use different tools, but also have different privileges on machines.

In contradiction to the previous attacker, categorized to be a slow scanner, 113.108.21.16 falls into the category of fast scanners. This scanner, as previously analysed scanners, uses a fixed IP identification value of zero. In addition to using an unconventional identification number, the scanner has sent the most packets, a total of 2,397,949, during the collection period, spread over 92 scans. Looking at those scans the number of packets sent per scan varies, however the time of these scans is between five and four seconds. The spread of the scans in combination with the large volume of probes sent over the entire period of time suggest that this scanner is performing some kind of hit and run.

Not only does the attacker perform some kind of hit and run, but the scans also seem to be cyclic as well. The period in between 97 of the scans can be called constant as the time in between 47 of them is close to 1,040 seconds and the other ones close to 48,555 seconds. This fixed scanning time and the intervals between scans suggest that the attacker is probing the internet in blocks and does so in a cyclic pattern.

The scanner scans in total eleven different destination ports. As with previous scanners, the distribution of probes per targeted ports is not uniform, showing that the scanner also has the possibility to manage the scanning resources. Analysing the data reveals that this scanner does not classify as advanced, in terms of obfuscation, as the scanned destination ports are not mixed during the scans. In addition the scanner only uses twenty source ports those are, as the destination addressees, incremented by one each time a new probe is sent. The behaviour is highly predictable compared to other scanners using random port to scan subsequent addresses.

This section has highlighted several different traditional classes of scanners. It shows that indeed horizontal and strobe scanners exist, and that another type of scanner exists targeting only each address and port once. It also acknowledges the existence of fast and slow scanners, along with the specification of different scan durations which is more a behavioural patter than a classification.

In addition some scanners exhibit highly cyclic behaviour when it comes to interval times or even number of packets sent per scan. Also cyclical behaviour can be seen in the header values such as source ports or destination addresses. In addition to the different and similar behaviour, distinction in the complexity of used tools from these behaviours. The next section will expand this analysis on multi sourced scanners.

SCANNERS USING MULTIPLE MACHINES OR SOURCE IP

Opposed to the previous section some attackers use several IP addresses to perform their scans. Previous work performed by Bhuyan et al. [17], a survey on port scanners and the detection methods, shows that a distinction is made between single and multi sourced scanners. However no additional characteristics are given, this is reflected by the classification of multi host scanners, which consists in distinguishing between many to one and many to many relationships. Although this is a valid classification, as it covers all possibilities, it does not provide much information other than the attacker to target ratio in terms of used and scanned addresses. There is no information in the number of ports scanned, neither in the size of the block of IP addresses used by the scanner. Such information might reveal some behavioural characteristics that can be used to classify the different types of scanners.

The first subsection, will briefly cover known multi host scanners which are research centres and known security companies. The second part will analyse unknown multi host scanners in order to find characteristics, to aid their classification.

KNOWN SCANNERS

Known scanners can be defined as scanners which are open about their activities, such as advertising their behaviour on websites. Akin scanners are often security companies or research institutions. Known instances are the University of Michigan [70], Scans.io [71], and Shodan [72], the latter being a search engine which enables users to browse for devices connected to the internet, by specifying the used ports, protocols, or even types of devices. Searching for information about University of Michigan and Scans.io about their scanning activities reveals that they both use large /34 block of IP addresses to send probes. The use of such large blocks suggests that they use multiple addresses even possibly adjacent to perform their scanning. In addition some of these friendly scanners advertise the used tools, it is known that University of Michigan uses a detectable tool such as ZMap in order to perform their scans. Also, Shodan admits that they use a similar tool to NMap in order to scan for open ports on the Internet. Thus other known scanners could be using similar tools in order to perform scans.

Institution name	IP block	IP addresses	Packets
University of Michigan	141.212.121.0/24	49	1,672,725
University of Michigan	141.212.122.0/24	217	8,074,897
Shadowserver	184.105.139.0/24	39	1,432,428
Shadowserver	184.105.247.0/24	46	2,274,658
NAGRA	185.35.62.0/24	243	9,36,401
Shadowserver	216.218.206.0/24	45	2,512,337
Project Sonar	71.6.216.32/27	28	776,133
Shadowserver	74.82.47.0/24	80	6,691,640 (4.36%)
Total	-	747 (0.07%)	24,371,219 (4.36%)

Table 9.4: Known ZMap scanners having IP identification number set to 54321

An investigation on whether other known scanners can be identified using these two simple criteria determined from the information found on the previously presented known scanners would be very appropriate. The first criteria are the adjacency of the IP addresses, namely the occurrence of large blocks of IP addresses performing scans. The seconds criteria are the usage of detectable scanning tools such as Masscan or ZMap.

Analysis performed in the previous sections would allow us to spot such scanners. In previous section, the detection of several scanning tools is shown which can be used to fulfil the second proposed criteria. In addition, during the used tool analysis the distance between consecutive IP addresses is calculated. Although the calculation of the distance between IP addresses only looks if it is bellow 256, the results should give a good indication of the presence of IP clusters in the scanners using that tool, complying to the first criteria.

Table 6.1 shows a promising subset of scanners that might contain some known scanners. First it shows that there is a large number of scanners using ZMap like software which only send packets having their IP identification number set to 54321, which is the standard value for ZMap. In addition 79.63% of the IP addresses have a distance lower than 256 indicating that there could be a high amount of neighbouring addresses. This selection of IP addresses should reveal some friendly scanners if the selected criteria is common to scanners using multiple hosts.

Looking at the data several clusters of adjacent IP addresses can be identified, as shown by table 9.4. The table shows that there are eight clusters identified, representing in total four known scanners. These knowns scanners represent 0.07% of the total addresses having sent probes to the collection network. Although the number of used addresses is low, the generated traffic is disproportionate to it as it represents 4.36% of all logged traffic.

Searching for other clusters of addresses for other detectable scanning tools only revealed one security company. Errata Security performed a scan using five IP addresses and advertises that the use a /24 block addresses to perform the scans. During the scan the company sent 84,593 packets, this amount is considerably less than previously mentioned scanners.

From the detected known scanners one can deduct that based on the two criteria, IP closeness and usage of the same scanning tool, it is possible to find scanners using multiple source addresses. In addition most of the scanners use one or multiple /24 IP block to performs scans. However, there are some differences to be noted in the appearance of the number of IP addresses per block in the dataset. Shadowserver is a good example, for three /24 blocks only 40 IP addresses are present in the collected data while for the fourth one 80 are recorded. This would suggest that the scanners are distributing their resources in such a manner that their IP addresses scan different part of the internet. This shows that multi host scanners, as single host scanners also manage their resources.

This crude analysis of known multi host scanners reveals that from a data perspective they can be detected using the two proposed criteria. In addition, the data presented in table 9.4 suggests that there are difference in number of addresses used and number of packets sent per address insinuating different behaviours amongst these types of scanners. This arises the question if there are any other characteristics which multi host scanners have. Therefore it should be interesting to see whether unknown multi source scanners can be analysed to see whether different behaviours exists as shown in previous section for single host scanners.

UNKNOWN SCANNERS

Unlike friendly scanners, scanners with malicious desires do not intend to reveal their identity. Therefore they are not likely to advertise their information coupling their IP address to a domain name serving a web page with their information. This increases the difficulty of finding such scanners, however the criteria applied in previous section show that from the collected data multi host scanners can be found.

As with the known scanners the focus lies on IP addresses close to each other. In addition, the usage of the same tools will indicates a high probability that the IP addresses are part of a multi host scanner. The first discovered cluster is contained in the range of 43.255.191.141.0/24, where eight (141, 161, 162, 163, 165, 166, 168, 170) addresses sent packets to the range of the TU Delft. All of the IP addresses belong to the same company renting servers and cloud services located in Hong Kong. The cluster of addresses is responsible for a total of 14,488,583 packets. All of the addresses within this cluster use a static IP identification number of 54,321 suggesting the use of ZMap as scanning tool.

From these two characteristics it is likely that this cluster is a multi host scanner. ZMap can spoof source addresses when performing scans, allowing the scanner to use a cluster of source addresses to perform scans. However it could also be that this is a coincidence meaning that different scanners have rented or compromised servers from the company. Therefore in order to confirm that this is indeed a single scanner using multiple hosts, other characteristics of the scans needs to be analysed. The first step is to look at the number of packets sent by each source address. An equal amount of packets sent would be a good indication of a scanner spreading the load over a number of IP addresses in order to stay under the radar. The number of packets sent by the eight addresses is not in the same range, however a clear division can be seen. Five of the eight have sent around 2,500,000 packets whereas three have only sent around 800,000 packets. This clear division in packets sent does not support either theories, it tends to create another concept defining two multi host scanners being active. The scanned destination ports may yield additional information, if the scanners are scanning the same ports or different subsets it is a strong indication of them working together or not. Unfortunately, all the addresses in the cluster scan for port 22, which is the most commonly scanned port therefore not confirming that this is a multi host scanner.

Besides using the same tool, being adjacent, and scanning the same there are no more simple characteristics indicating this is a multi host scanner. However looking at another aspect, such as the time intervals the scan took place, similarities can indicate the cluster working together. The data shows that the addresses all scanned continuously during the recording period. In addition all the IP addresses in the cluster have scanned the entire range of the TU Delft. Each of the addresses having targeted around 58,000 unique IP addresses, the slight difference in scanned addresses can be explained by the TU assigning certain IP addresses, resulting in packets sent to certain destination IP addresses being recorded in the data at one point and not at another point in time.

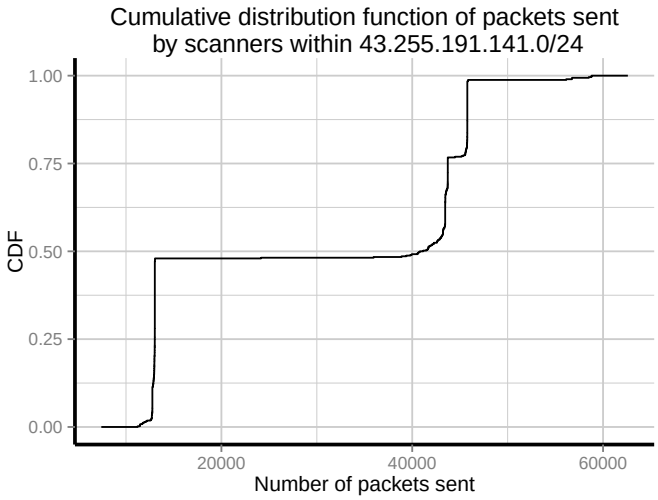


Figure 9.3: CDF of number of packets sent per source IP of cluster 43.255.191.141.0/24

The scans performed by the cluster, as defined in previous section, shows that each of the IP addresses have performed scans of 227 seconds. Also the number of packets sent during these scans is close to one of three values for all the source addresses, this is quite well pictured by figure 9.3. The figure shows that almost 50% of the performed scans had a packet rate of 13,000, 25% of 43,000, and 25% of 46,000 packets. Now it could be that some of the IP addresses send an amount of packets per scan and the other addresses the other amount. However the data shows that each source IP address performs scans sending the three defined number of packets. This further indicates that these IP addresses belong to a multi host scanner.

Combining the several described characteristics leads to the conclusion that this cluster is indeed a multi host scanner. Finally an additional conclusion can be made which introduces a new subclass to multi host scanners. During the entire scanning period, none of the performed scans is intersecting the others more explicitly none of the scans are performed simultaneously. This in combination with the used software ZMap, which allows for IP spoofing, shows that the user is using a single physical machine. Using a single machine is the most probable scenario as syncing different machines to perform non-intersecting scans is seems to be a tedious amount of work for what can be achieved with a scanning tool as ZMap.

Another cluster of IP addresses in the 94.31.49.0/24 is also suspected to be operated by a multi host scanner. In total 116 IP addresses from this subnet perform scans towards the network of the TU Delft. Although the attacker uses significantly more IP addresses than previous analysed scanner, he only sends 164,419 packets in total. All the IP addresses are registered to the Zayo Group, which is also a cloud service provider. Similar to the previous case, the tool used by the different source addresses is also ZMap like as all the IP identification numbers are statically set to 5,4321.

Source port	Packets	Source IPs
21	28,189	79
22	33,157	81
23	16,409	82
443	33,870	92
995	34,015	90
5222	18,291	83
10000	488	3

Table 9.5: Port scanned by 94.31.49.0/24

Contrary to the previous scanner, this one scans for different destination ports. Table 9.5 shows initially not all ports are scanned equally, namely destination port 10,000 and 23 received less traffic. A simple explanation can be given for port 10,000 which is the last port scanned, therefore increasing the likelihood that the collection period has ended during the probing of that port.

Figure 9.4 shows that port 10,000 is indeed scanned last and that the scanning period was interrupted, confirming that each port, excepted from 23, has been scanned with the same intensity. The figure also reveals that all the other ports are scanned in a single time frame. This confirms that this cluster is part of a multi host scanner as it is unlikely that separate scanners all scan the same port (which are not all popular destination ports) during the same time period.

The figure reveals more information, each port is scanned for the same duration, this is reflected by the number of source addresses scanning the different ports, laying between 79 and 90 addresses. In addition the number of packets sent by each source address which varies between 1,000 and 2,000 packets. The uniformity of scanning combined with the fact that port 23 received less traffic, shows that the scanner is capable of managing resource, preferring some ports over others.

Besides managing his resources, this scanner is, contrary to the previous scanner using multiple physical machines. It is clear that the scanner is using ZMap, however the tool does not allow for simultaneous port scanning [73]. It is not possible to run multiple instance of the software on one machine as it may cause interference. However the scanner scans two different ports (23 and 5222) during the same time frame. Taking a closer look shows us that packets destined for different ports are simultaneously received from two addresses within the cluster. This is not possible when using a single machine, as ZMap does not allow for simultaneous port scanning, confirming that the scanner using multiple addresses also employs multiple machines.

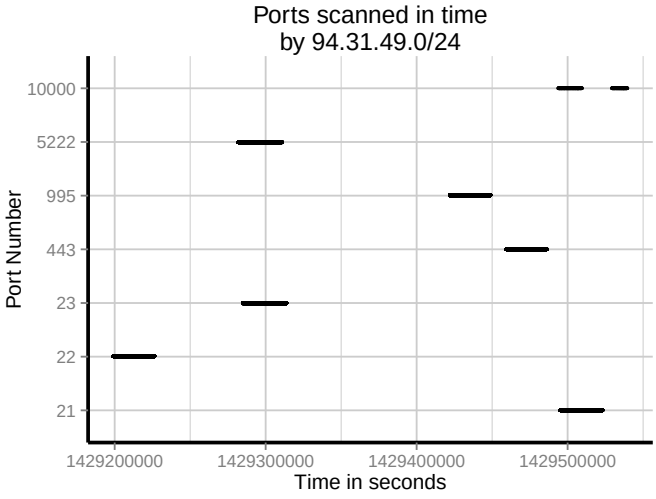


Figure 9.4: Plot of scanners showing the umber of unique IP and unique ports scanned

The final cluster of IP addresses being analysed belong to the 115.160.65.0/24 cluster. The group of IP addresses belong to the to Seokyung Cable Television an ISP located in South Korea. All 12 IP addresses send

packets which all have the same fixed sequence and IP identification numbers. Although the use of a fixed IP identification number suggests a ZMap like tool, the use of a static sequence number indicates that the attackers uses a different tool or a modified version of it.

In total, the cluster performs 327 scans during which the addresses sent 10.719 packets. Striking about this scanner is the fact that contrary to previous scanners, it is a slow scanner. Every address within the cluster sends at a rate of less than one packet every hundred seconds. Paradoxical to the previous scanner, this cluster does not have a uniform scanning time nor a number of packets sent per performed scan.

As with previous scanners, this cluster scans one port more often than the others. Table 9.6 shows that port 23 is probed more intensively than the other ports. The varying probing intensity suggest that this scanner also manages his resources, which is confirmed by the scans of ports not being cut off at the end or beginning of the data collection. In addition each source address scans each port and only scans a destination IP once (for all ports except 23), suggesting that the scanner does not want to waist resources by sending additional packets. The resource management is also confirmed by the fact that several source addresses are scanning the same destination ports but that the same address is never scanned twice on the same port. This shows that the addresses used by the scanner complement each other in terms of destination address and ports.

Source port	Packets	Destination IPs
22	398	398
23	8,765	8,126
8,080	768	768
32,764	565	565
58,455	223	223

Table 9.6: Port scanned by 115.160.65.105/24

Finally this cluster also shows similar behaviours as previously analysed single source scanners. I.e. the destination addresses are scanned in a particular order. Each source address increments the destination address by 255. In addition there exists a correlation between destination address, destination port and source port. Each time the destination address range is changed, meaning the last three digits changing, the source port also changes from which the address is scanning. This similar behaviour and characteristics of single sourced scanners and multi sourced scanners, suggest that this ZMap like tool can be used by both multi- and single-host scanners.

These three examples of multi host scanning shows that the classic categorization of scanners also holds for them, there are horizontal and strobe scanners. In addition multi host scanners are also able to manage their resources, by probing some ports more often than others. More interesting is that the multi host scanners also have a subdivision, there are scanners which only use one physical / virtual machine and other which at least use two. In the next section a discussion will combine the different findings in order to present a full overview of the performed work.

10

FINAL THOUGHTS

This final chapter is a concise argumentation reviewing all anterior findings that have been presented in the previous chapters of this thesis. Following this discussion a conclusion is made in which the most important aspects are summarized and directions of future work are suggested.

DISCUSSION

This discussion explains and dissects the findings presented in previous chapters. The first performed analysis focuses on the data. Throughout this analysis the data is shown to be unbiased because all of the IP addresses in the collection range received an equal amount of packets. Assuming that the scanners have a uniform distribution of targets, as done by Durumeric et al. [29]. In addition to representative data, the results of the initial analysis are similar, if not a duplicate of the results presented by Durumeric et al. This is an interesting results as the dataset used for this thesis is smaller than the dataset used in previously mentioned research. The dataset used by Durumeric et al. is an unused /8 block, whereas this dataset is formed from two partially used /24 networks. In addition the duration of data collection is also different, the present dataset represents a month of data whereas the other research uses a years worth. This shows that for simple analytic such as number of packets from a certain protocol or destined to a certain port can be performed on a smaller scale.

The general analysis of used scanning techniques and scanning tools also add to get a better overview of the scanning behaviour. From the collected data it is obvious that the TCP SYN scan is the preferred method for scanning as they form 87.91% of the received TCP packets. This can be explained by the fact that it is the fastest method of scanning and is a current option in most known scanning tools. However the usage of this scanning method also requires root privileges as the standard TCP and IP implementation of the machine is bypassed. This suggests that most of the scanners have full control over the machines used for scanning.

In addition to the used scanning technique simple analysis of the header values in both the TCP and IP protocol for the TCP SYN packets show that some values are preferred over others. As the values should be randomly chosen, the numbers in the headers should be uniformly distributed amongst their respective ranges. However the data clearly shows that this is not the case indicating canning tool characteristics. Five TCP sequence numbers are found in 2.17% of all packets. This general observation is also reflected in other header values such as the port from which the packet was sent. Five source ports generate 34.19% of the traffic besides a clear difference can be seen in the amount of packets sent from two ranges within the range of the source ports. Remarkably, during the research source ports which lay in the system range (0-1024) are used for scanning, confirming that some scanner do have root privilege on their machines. It is also an indication that some scanner might use these ports to bypass firewalls restricting them to send outgoing packets. The bypassing behaviour was noted for a scanner using a web server to perform scans from port 80 thereby covering his scans as regular traffic. The analysis of the header information shows that there are some preferred setting or tools used for scanning, which manifest in the header values.

Further analysis reveals that the scanners may well be using specific tools to perform the scans. This is

not a new finding as Durumeric et al [29] already describes two scanning tools ZMap and Masscan which can be detected. They also present statistics which reflecting the analysis of this dataset. However they only identify scanners using 5,4321 for IP identification as using ZMap. This can be elaborated to a more general class, such as scanners using a fixed IP identification number as suggested by the IP header analysis.

In addition to Masscan and ZMap, Unicorn scan is also a detectable scanning tool. However the number of scanners using it is significantly lower than the other tools. This might be due to the fact that the software is dated, and the website hosting the software is down at the moment of writing.

What becomes obvious from the analysis of the tools is that they all store information into the sent package. The information is often put into the sequence number, which allows these tools to recognize packets responses to scans. This is an important feature as it allows the tools to perform fast TCP SYN style scanning considering they do not need to wait for a response form the target. Also the software does not need to put the scanned targets in memory as response packages can be filtered out of the regular traffic. Resulting in fast and lightweight scanning software.

The method using sequence numbers as filter for scanning packets is also applied by simple scanners. From the TCP header analysis it becomes clear that some sequence numbers are preferred over others. This is also reflected in the type of scanner used. Some scanners only send packets using a static sequence number, which is a more crude technique for recognizing responses from scanned targets than used by previously discussed scanning software. Demonstrates that not only do scanners use a variety of known tools but also develop their own tools, and that the sophistication of these tools widely differs.

Besides noticing a difference in used tools, it has become evident that the geographical location of the machines used to perform scans is not equally distributed. This was already noted by Lee et al. **??** published in 2003. In the article the authors describe the usage of a /8 size network to perform the research leading to similar results presented in this thesis. This validates that smaller datasets contain the same amount of information. However the conclusion of the authors does not correlate with the findings, they claim that the distribution of scanners is only correlated with the availability of internet connections. This is not supported by the data, as the top 10 countries scanning, corrected by several factors as the number of available connections fluctuates. The fluctuation indicates that there are other factors in play, however it remains that there are countries popping up in the top ten for several corrections. This indicates that some countries are preferred over other when it comes to scanning activity. The most obvious explanation would be indeed the available connections, but legislation may also play a significant role.

Besides a geographical analysis a time analysis also reveals valuable information. As basic packets per hour analysis shows that during the recording period there where some peeks of scanning activities. From those peaks one could be attributed to a single host performing an intense and short scan. The second peak is being cause by a larger group of IP addresses. The difference in received packets is a good indication of different types of scanners. This assumption is strengthened by the observation of different peaks in received traffics when looking at an hourly scale.

The elementary analysis suggests that there are different types of scanners as defined in previous studies, namely slow and fast scanners. From those two categories we can determine that a slow scanner will have to scan for a longer period of time to cover the same amount of targets than a fast scanner would. However in current literature no exact information is given on how long scanners scan. From the data we can see that the majority of scanners scan for a short period of time, more than 90% have only sent packets on less than three separate days. This would suggest that the majority of scanners could be considered as fast, which is confirmed by the interval time of packets, which is mostly below 10 seconds. However this is where using a small dataset could introduce some bias, as there are only two /24 addresses to scans. Depending on the definition of slow scanner, a scanner would not need more than three days to perform a scan spanning such a short range of IP addresses, therefore biasing the ratio between slow and fast scanner. Nevertheless, the data clearly shows that the difference between slow and fast scanners exists. This is confirmed by the distribution of scan duration showing that the majority of scans are short, but that there are also a considerable amount of long scans.

The classification of scanners, can not only be done by number of packets sent in a time period, thereby classifying them as slow or fast, it can also be done by the targeted addresses and ports as described in the article written by Bhuyan et al. [17]. The first distinction made between scanners is based on their targets, which can be classified as vertical, and strobe scanners. Although this classification covers all possibilities,

there is an additional class observed during this study, this being the category scanning a target only once. This means that the scanner never scans the same IP twice and never scans the same port twice on different IP addresses.

As shown by this research these classifications, which are often attributed to single source scanners, also hold for multi source scanners. Multiple IP clusters, both known and unknown exhibit scanning behaviour that can be attributed to one of those classes. Therefore there is no real difference in terms of classification between single source and multi source scanner other than the recorded amount of IP addresses used to perform the scanning.

In addition to the same classes, the multi source and single source scanners share other characteristics. From their analysed behaviours using different metrics such as scan time, correlation between values found in the headers of the packets, the multi source and single sourced scanners show similarities. The first striking behaviour is the management of resources. In both cases, single and multi source, the scanners are able to scan some ports more frequently than others. This behaviour suggests that they are more interested in finding some ports opened then others.

Another shared behaviour is some kind of cyclic behaviour when it comes to targeted addresses. In both the single source and multi sourced scanners, IP addresses are incremented by 255 instead of being random or simply incremented by one. This shows that some scanners are more concerned for detection systems than others. When scanning IP addresses sequentially it is more likely that an intrusion detection system will flag the attacker. This is due to the nature of IP allocation, which is always performed in blocks of adjacent IP addresses. By incrementing the next targeted IP address by 255 instead of one, the chances of detection decrease as the next scanned IP address is likely assigned to another network with a separate scan detection system.

The final share of behaviours is more complex. Scanners from both the single and multi source class tend to show some form of cyclic or fixed behaviour. In both classes, scanners tend to send a fixed amount of packets per scan, or have a fixed amount of time in between scans. Also correlations have been noted between certain header values. These behavioural tendencies can be used not only to confirm if a scanner is a single host or multi host scanner but additionally to detect them.

CONCLUSION

In the discussion following the analysis of the data some remarkable findings are made. First it shows that a relatively small dataset can be used to perform scanning analysis yielding the same results as analysis over larger datasets. This is interesting because it allows for more institutes, with a smaller collection range to perform such research. Meaning that with these smaller dataset collaboration can be performed in order to compare the scanned range of scanners or to see whether their behaviour is comparable in different subnets.

Secondly, small datasets can reveal interesting information when analysed from a dataset point of view. Analysing the dataset for anomalies resulted in the finding of new scanning tools. These tools use fixed header values that should be random. In addition to these new tools, the header values also show that the tools used by scanners vary in terms of complexity. Some scanning tools use fixed values for some fields whereas some use modular fields. Also some tools are used more often than others, are used to scan different port, and some even seem to be custom made.

Thirdly, the discussion and the presented work also concludes that the defined formula to calculate the scanning time works. It allows showing similarities and constant value for some scanners, which leads to the fourth point cyclic behaviours.

Fourthly, the discussion also ascertains that some scanners do show some cyclic behaviours. These cyclic behaviours are not limited to one header value property and are shared by some single host scanners. Such cyclic behaviours have been defined on the time scale, showing that scanners perform scans of the same length or number of packets. This cyclical characteristic is also confirmed by some scanners on the destination address scale where consecutive packets are sent to consecutive addresses. This behaviour can not only be used to compare single host scanners but can also define multi host scanners.

Finally, the work has defined a new class of scanners. Scanners only scanning a destination and source port once. This has been seen for single source scanners, however during the preliminary research, this behaviour has been recorded for a multi sourced scanner using a XMAS scanning style. This type of new scanner is indeed more appropriate for a multi sourced scanner as a for a single sourced scanner it would reveal no valuable information. Hence rises the the question whether the single sourced scanners using this scanning technique are not part of a multi sourced scanner of which the source addresses lie further apart, thus making

it seem to be single sourced scanners. The raised question leads to the final part of the thesis: future work, presenting questions and proposing further research topics which arose during the writing.

FUTURE WORK

As mentioned in the final part of the conclusion, the possibility of multi host scanners using distant addresses does exist. A challenging investigation could be conducted, verifying the existence of these types of scanners. This could be achieved by taking all the scanners scanning one destination port and address and comparing their scanned targets. If the targets scanned by these scanners does not intersect the probability of them working together increases. In addition the used tools and similarities in behaviour will also increase the probability of the hosts working together.

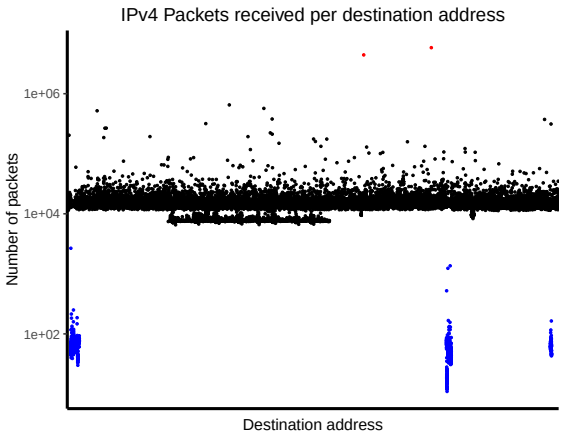
Further research could focus on the implementation of behavioural findings presented in the thesis in known scanner detection tools. Tools such as Snort [74] and Suricata [75] allow users to add rules. These rules could be defined in such a way that scanners exhibiting behaviours presented in the thesis can be detected.

Another interesting work would be to classify scanners in terms of scanning speed and probed targets. As mentioned in both the discussion and conclusion, the data collected reflects the dataset of /8 networks. This means that datasets of size /24 can be used to monitor different part of the internet and thus also collect different metrics of scanners scanning different address ranges. Comparing the number of packets sent per scan can reveal if scanners are scanning at a steady pace of that they vary the intensity. It can also reveal if scanners are scanning the entire Internet at once or only parts of it. Also the combination of several small datasets might reveals the order in which parts of the internet are scanned. This thesis has shown that scanners show cyclic behaviours, these behaviours can be monitored across different ranges.

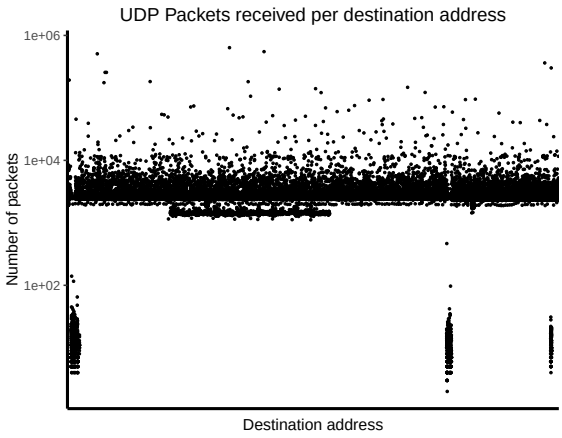
This thesis illustrates that the detection of NMap is possible. But the results shown in the thesis indicated that detecting this used software remains challenging as the absolute number of detected scanners is low. The use of a larger test network might give additional information on some other characteristics with NMap, which when combined with the known signature can be used to identify more scanners using the software.

A

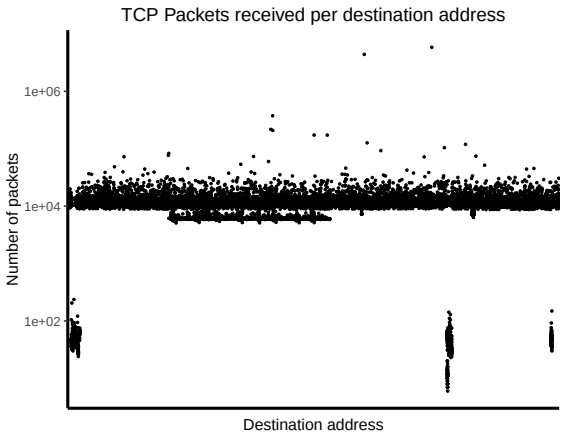
PACKET TYPE RECEIVED PER ADDRESS



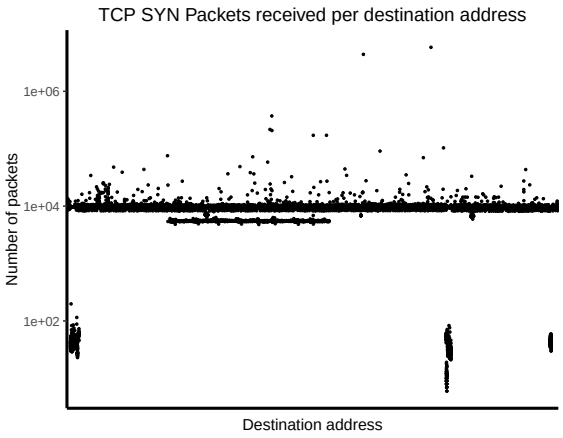
(a) Number of IPv4 packets received by destination IP



(b) Number UDP of packets received by destination IP



(c) Number of TCP packets received by destination IP



(d) Number of TCP SYN packets received by destination IP

Figure A.1: Number of packets received per IPv4 destination address

BIBLIOGRAPHY

[1] J. Postel, *Internet Protocol*, STD 5 (RFC Editor, 1981) http://www.rfc-editor.org/rfc/rfc791.txt.

[2] J. Postel, *User Datagram Protocol*, STD 6 (RFC Editor, 1980) http://www.rfc-editor.org/rfc/rfc768.txt.

[3] J. Postel, *Transmission Control Protocol*, STD 7 (RFC Editor, 1981) http://www.rfc-editor.org/rfc/rfc793.txt.

[4] M. Cotton, L. Eggert, J. Touch, M. Westerlund, and S. Cheshire, *Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry*, BCP 165 (RFC Editor, 2011).

[5] ICAAN, *Remaining ipv4 addresses to be redistributed to regional internet registries | address redistribution signals that ipv4 is nearing total exhaustion,* (2014), https://www.icann.org/news/announcement-2-2014-05-20-en.

[6] J. Research, *'INTERNET OF THINGS' CONNECTED DEVICES TO ALMOST TRIPLE TO OVER 38 BILLION UNITS BY 2020*, Tech. Rep. (Juniper Research, 2015) http://www.juniperresearch.com/press/press-releases/iot-connected-devices-to-triple-to-38-bn-by-2020.

[7] J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister, *Census and survey of the visible internet,* in *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement* (ACM, 2008) pp. 169–182.

[8] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, *A search engine backed by Internet-wide scanning,* in *Proceedings of the 22nd ACM Conference on Computer and Communications Security* (2015).

[9] G. Lyon, *Nmap,* (1997), https://nmap.org/.

[10] Z. Durumeric, E. Wustrow, and J. A. Halderman, *Zmap: Fast internet-wide scanning and its security applications.* in *Usenix Security* (2013) pp. 605–620.

[11] MITRE, *CVE-2014-7169.* Available from MITRE, CVE-ID CVE-2014-7169. (2014), https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271.

[12] B. Delamore and R. K. Ko, *A global, empirical analysis of the shellshock vulnerability in web applications,* in *Trustcom/BigDataSE/ISPA, 2015 IEEE*, Vol. 1 (IEEE, 2015) pp. 1129–1135.

[13] G. Kambourakis, T. Moschos, D. Geneiatakis, and S. Gritzalis, *Detecting dns amplification attacks,* in *Critical Information Infrastructures Security* (Springer, 2008) pp. 185–196.

[14] A. Mairh, D. Barik, K. Verma, and D. Jena, *Honeypot in network security: a survey,* in *Proceedings of the 2011 International Conference on Communication, Computing & Security* (ACM, 2011) pp. 600–605.

[15] R. JAŠEK, M. KOLAŘÍK, and T. VÝMOLA, *Apt detection system using honeypots,* in *Proceedings of the 14th WSEAS International Conference on Automation & Information (ICAI'13). Montreux: WSEAS Press* (2013) pp. 25–29.

[16] W. El-Hajj, F. Aloul, Z. Trabelsi, and N. Zaki, *On detecting port scanning using fuzzy based intrusion detection system,* in *Wireless Communications and Mobile Computing Conference, 2008. IWCMC'08. International* (IEEE, 2008) pp. 105–110.

[17] M. H. Bhuyan, D. Bhattacharyya, and J. K. Kalita, *Surveying port scans and their detection methodologies,* The Computer Journal , bxr035 (2011).

[18] A. S. Tanenbaum, *Computer networks. 4th* (Prentice Hall PTR, 2002).

[19] Internet Society, *Rfc editor,* https://www.rfc-editor.org/.

[20] Internet Engineering Task Force, *Internet engineering task force,* https://www.ietf.org/.

[21] K. Ramakrishnan, S. Floyd, and D. Black, *The Addition of Explicit Congestion Notification (ECN) to IP*, RFC 3168 (RFC Editor, 2001) http://www.rfc-editor.org/rfc/rfc3168.txt.

[22] K. Lougheed and Y. Rekhter, *Border Gateway Protocol 3 (BGP-3)*, RFC 1267 (RFC Editor, 1991) http://tools.ietf.org/html/rfc1267.

[23] U. Kanlayasiri, S. Sanguanpong, and W. Jaratmanachot, *A rule-based approach for port scanning detection,* in *Proceedings of the 23rd electrical engineering conference, Chiang Mai Thailand* (2000) pp. 485–488.

[24] J. E. Dickerson, J. Juslin, O. Koukousoula, J. Dickerson, *et al.*, *Fuzzy intrusion detection,* in *Ifsa world congress and 20th nafips international conference, 2001. joint 9th*, Vol. 3 (IEEE, 2001) pp. 1506–1510.

[25] S. M. Bridges and R. B. Vaughn, *Fuzzy data mining and genetic algorithms applied to intrusion detection,* in *Proceedings of 12th Annual Canadian Information Technology Security Symposium* (2000) pp. 109–122.

[26] L. Ertoz, E. Eilertson, A. Lazarevic, P.-N. Tan, P. Dokas, V. Kumar, and J. Srivastava, *Detection of novel network attacks using data mining,* in *Proc. of Workshop on Data Mining for Computer Security* (Citeseer, 2003).

[27] M. Dabbagh, A. J. Ghandour, K. Fawaz, W. Hajj, and H. Hajj, *Slow port scanning detection,* in *Information Assurance and Security (IAS), 2011 7th International Conference on* (IEEE, 2011) pp. 228–233.

[28] C. B. Lee, C. Roedel, and E. Silenok, *Detection and characterization of port scan attacks,* Univeristy of California, Department of Computer Science and Engineering (2003).

[29] Z. Durumeric, M. Bailey, and J. A. Halderman, *An internet-wide view of internet-wide scanning,* in *USENIX Security Symposium* (2014).

[30] G. Combs, *Tshark—dump and analyze network traffic,* (2012), https://www.wireshark.org/docs/man-pages/tshark.html.

[31] V. Jacobson, C. Leres, and S. McCanne, *libpcap, lawrence berkeley laboratory, berkeley, ca,* Initial public release June (1994), https://wiki.wireshark.org/Development/LibpcapFileFormat.

[32] L. Florio and K. Wierenga, *Eduroam, providing mobility for roaming users,* in *Proceedings of the EUNIS 2005 Conference, Manchester* (2005).

[33] Cisco, Cisco (2008), http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/10559-22.html.

[34] D. C. Plummer, *Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware*, STD 37 (RFC Editor, 1982) http://www.rfc-editor.org/rfc/rfc826.txt.

[35] J. Postel, *Internet Control Message Protocol*, STD 5 (RFC Editor, 1981) http://www.rfc-editor.org/rfc/rfc792.txt.

[36] A. Conta and S. Deering, *Generic Packet Tunneling in IPv6 Specification*, RFC 2473 (RFC Editor, 1998) http://www.rfc-editor.org/rfc/rfc2473.txt.

[37] E. Wustrow, M. Karir, M. Bailey, F. Jahanian, and G. Huston, *Internet background radiation revisited,* in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (ACM, 2010) pp. 62–74.

[38] G. Combs *et al.*, *Wireshark,* (2007), https://www.wireshark.org.

[39] IANA, *Service name and transport protocol port number registry,* (2016), http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt.

[40] G. Camarillo, *The Internet Assigned Number Authority (IANA) Uniform Resource Identifier (URI) Parameter Registry for the Session Initiation Protocol (SIP)*, Tech. Rep. (2004) http://www.iana.org/.

[41] T. Yeh, *Netis routers leave wide open backdoor,* Available from Trend Micro (2014), http://blog.trendmicro.com/trendlabs-security-intelligence/netis-routers-leave-wide-open-backdoor/.

[42] C. Beedgen, *Malware faq: Microsoft windows upnp vulnerabilities,* Available from SANS, https://www.sans.org/security-resources/malwarefaq/win_upnp.php.

[43] D. Geneiatakis, T. Dagiuklas, G. Kambourakis, C. Lambrinoudakis, S. Gritzalis, S. Ehlert, D. Sisalem, *et al.*, *Survey of security vulnerabilities in session initiation protocol.* IEEE Communications Surveys and Tutorials **8**, 68 (2006).

[44] S. M. Specht and R. B. Lee, *Distributed denial of service: Taxonomies of attacks, tools, and countermeasures.* in *ISCA PDCS* (2004) pp. 543–550.

[45] R. Beverly, *A robust classifier for passive tcp/ip fingerprinting,* in *Passive and Active Network Measurement* (Springer, 2004) pp. 158–167.

[46] M. Zalewski, *p0f v3* (2014), http://lcamtuf.coredump.cx/p0f3/.

[47] O. Arkin, *Icmp usage in scanning,* Black Hat Briefings (2000).

[48] M. De Vivo, E. Carrasco, G. Isern, and G. O. de Vivo, *A review of port scanning techniques,* ACM SIGCOMM Computer Communication Review **29**, 41 (1999).

[49] D. Myers, E. Foo, and K. Radke, *Internet-wide scanning taxonomy and framework,* in *Proceedings of Australasian Information Security Conference (ACSW-AISC), 27-30 January 2015* (Australian Computer Society, Inc, 2015).

[50] D. Adrian, Z. Durumeric, G. Singh, and J. A. Halderman, *Zippier zmap: internet-wide scanning at 10 gbps,* in *8th USENIX Workshop on Offensive Technologies (WOOT 14)* (2014).

[51] J. Daemen and V. Rijmen, *The design of Rijndael: AES-the advanced encryption standard* (Springer Science & Business Media, 2013).

[52] J.-P. Aumasson and D. J. Bernstein, *Siphash: a fast short-input prf,* in *Progress in Cryptology-INDOCRYPT 2012* (Springer, 2012) pp. 489–508.

[53] R. E. Lee and J. C. Louis, *Unicornscan - history, background and technical details,* (Presented at Defcon 13, 2005).

[54] G. F. Lyon, *Nmap network scanning: The official Nmap project guide to network discovery and security scanning* (Insecure, 2009).

[55] G. Lyon, *Nmap decode function,* (1997), https://github.com/nmap/nmap/blob/799048e9fc576091623f9ce5f79e26d80020f98b/scan_engine_raw.cc.

[56] G. Lyon, *Nmap decode function,* (1997), https://github.com/nmap/nmap.

[57] L. D. Brown, T. T. Cai, and A. DasGupta, *Interval estimation for a binomial proportion,* Statistical science , 101 (2001).

[58] C. J. Clopper and E. S. Pearson, *The use of confidence or fiducial limits illustrated in the case of the binomial,* Biometrika **26**, 404 (1934).

[59] R. Munroe, *Map of the internet,* (2006), http://xkcd.com/195/.

[60] C. Botnet, *Internet census 2012: Port scanning/0 using insecure embedded devices,* URL http://internetcensus2012. bitbucket. org/paper. html (2013).

[61] L. MaxMind, *Maxmind geoip country database,* (2007), https://www.maxmind.com/en/home.

[62] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye, *Ip geolocation databases: Unreliable?* ACM SIGCOMM Computer Communication Review **41**, 53 (2011).

[63] IANA, *Autonomous system (as) numbers,* (2016), http://www.iana.org/assignments/as-numbers/as-numbers.xhtml.

[64] J. A. Hartigan, *Clustering algorithms* (1975).

[65] L. v. d. Maaten and G. Hinton, *Visualizing data using t-sne,* Journal of Machine Learning Research **9**, 2579 (2008).

[66] V. Ghiette, N. Blenn, and C. Doerr, *Remote identification of port scan toolchains,* in *2016 8th International Conference on New Technologies, Mobility and Security (NTMS)* (IEEE, 2016).

[67] S. Jones, *Port 0 os fingerprinting,* Network Penetration, viewed Aug **22**, 2005 (2003).

[68] D. Miessler, *Not All SYNs Are Created Equal*, Tech. Rep., https://danielmiessler.com/study/synpackets/.

[69] S. Ganti, *MyDoom and its backdoor*, Tech. Rep. (SANS Institute, 2004).

[70] University of Michigan, *Proxy test,* (2016), http://proxytest.zmap.io/.

[71] Scans.io, *Internet-wide scan data repository,* (2016), https://scans.io/.

[72] J. C. Matherly, *Shodan the computer search engine,* Available at [Online]: http://www. shodanhq. com/help (2009), https://www.maxmind.com/en/home.

[73] ghost, *Zmap issue #44,* (2013), https://github.com/zmap/zmap/issues/44.

[74] M. Roesch *et al., Snort: Lightweight intrusion detection for networks.* in *LISA*, Vol. 99 (1999) pp. 229–238, https://www.snort.org/.

[75] I. Suricata, *Open-source ids/ips/nsm engine,* (2014), https://suricata-ids.org/.